P. 1877/92

11 1992

informatyka

WYDAWNICTWO SIGMA NOT

Cena 18 000 zł

informatyka

nr 11 1992
Listopad
Rok wydania XXVII

Miesięcznik
ISSN 0542-9951
INDEKS 36124

KOLEGIUM REDAKCYJNE:

mgr Jarosław DEMINET
mgr inż. Piotr FUGLEWICZ
mgr Teresa JABŁOŃSKA
(sekretarz redakcji)
Władysław KLEPACZ
(redaktor naczelny)
dr inż. Wojciech MOKRZYCKI
mgr inż. Jan RYZKO
dr Zdzisław SZYJEWSKI
mgr Hanna WŁODARSKA – MARCZENKO

PRZEWODNICZĄCY
RADY PROGRAMOWEJ:

Prof. dr hab.
Juliusz Lech KULIKOWSKI

WYDAWCA:
Wydawnictwo Czasopism i Książek Technicznych SIGMA NOT
Spółka z o.o.
ul. Ratuszowa 11
00-950 WARSZAWA
skrytka pocztowa 1004

Redakcja:
01-552 Warszawa,
Pl. Inwalidów 10, p. 104, 105
tel. 39-14-34

Materiałów nie zamówionych redakcja nie zwraca

W sprawach ogłoszeń prosimy zwracać się bezpośrednio do Redakcji lub Działu Reklamy i Marketingu
00-950 Warszawa
ul. Mazowiecka 12
telefon: 27-43-66
telefaks: 19-21-87
teleks: 814877

W numerze:

	Strona
AMIL – interakcyjny język symulacji układów dynamicznych – *Marek M. Stabrowski*	1
Interakcyjny system do projektowania i analizy cyfrowych układów logicznych – *Piotr Bizacki, Grażyna Paliwoda, Marek Skomorowski*	5
DIANA – 9 – pakiet wspomaganej komputerowo analizy diagnostycznej i projektowania struktur organizacyjnych – *Edward Michalewski*	9
Diagnostyka systemów informacyjnych (1). Problemy podstawowe – *Agnieszka Szewczyk*	12
Dlaczego NIS? – *Rafał Krajewski*	19
Dydaktyka informatyki w uczelniach rolniczych – *Jadwiga Orylska*	22
Środowisko systemów CAD – *T. Winkler*	25

W najbliższych numerach:

- **Jerzy Surma przedstawia zarys najbardziej istotnych aspektów modelowania wiedzy w metodyce KADS.**
- **Zdzisław Wrzeszcz opisuje sposób realizacji funkcji wyszukiwania w bazie danych systemu GFG.**
- **Bogusław Jackowski i Marek Ryćko zajmują się regułami dzielenia (przenoszenia) wyrazów w języku polskim przy składzie komputerowym.**
- **Artur Chojnacki i Zbigniew Kalisiak charakteryzują uniwersalny monitor szyny dla dydaktycznego systemu mikroprocesorowego.**
- **Grzegorz Kowalczyk zajmuje się sieciowym systemem plików NFS umożliwiającym współpracę w sieci lokalnej komputerów o różnej architekturze.**
- **Mirosław Szczypiński podaje rozwiązanie problemu jednoczesnej pracy systemów NetWare i UNIX z użyciem pakietu PC NFS.**

Warunki prenumeraty

Przyjęcie prenumeraty – wyłącznie na podstawie dokonanej wpłaty na drukach dostarczanych dotychczasowym prenumeratorom przez Wydawnictwo, lub nowym – po uprzednim zgłoszeniu zapotrzebowania (pisemnie lub telefonicznie) w Zakładzie Kolportażu Wydawnictwa.

Blankiet wpłaty – powinien zawierać następujące informacje: dokładna nazwa i adres (z kodem pocztowym) zamawiającego, tytuły zamawianych czasopism, ich liczbę i okres prenumeraty.

Wpłata – zgodnie z podanymi cenami należy dokonać w banku lub w UPT na konto podane na naszym blankiecie, tj:
Państwowy Bank Kredytowy III O/Warszawa nr: 370015-1573-139-11

Prenumeratorzy zbiorowi – osoby prawne obowiązują blankiety „Wpłata-Zamówienie". Cena normalna.

Prenumeratorzy indywidualni – osoby fizyczne obowiązują blankiety typu przekazy dla wpłat na rachunki bankowe. Cena normalna.

Prenumerata ulgowa – zgodnie z podaną ceną ulgową przysługuje wyłącznie osobom fizycznym, będącym członkami SNT, studentom i uczniom szkół zawodowych. Uczniowie szkół ogólnokształcących mogą zamówić w prenumeracie ulgowej tylko miesięcznik „Aura"

Uwaga! w podanym okresie prenumeraty można zamówić tylko po jednym egzemplarzu z każdego tytułu.

Prenumerata ze zleceniem wysyłki za granicę – cena prenumeraty ze zleceniem wysyłki za granicę jest dwukrotnie wyższa od ceny normalnej.

Należy podać dokładny adres odbiorcy za granicą.

Terminy przyjmowania prenumeraty:
– do 10 listopada na I, II, III, IV kwartał następnego roku
– do 28 lutego na II, III, IV kwartał br.
– do 31 maja na III i IV kwartał br.
– do 31 sierpnia na IV kwartał br.

Zmiany w prenumeracie, np. zmiana liczby tytułów, liczby egzemplarzy, rezygnacja z prenumeraty, można zgłaszać tylko w podanych terminach z mocą obowiązującą od następnego kwartału.

Egzemplarze archiwalne (z lat ubiegłych)
Można nabyć za gotówkę w Klubie Prasy Technicznej, Warszawa, ul. Mazowiecka 12 (tel. 26-80-16) lub zamówić pisemnie w Zakładzie Kolportażu, Dział Handlowy, 00-950 Warszawa, skr. poczt. 1004 (tel. 40-37-31), na rachunek lub za zaliczeniem pocztowym.

Informacji o prenumeracie udziela: Zakład Kolportażu Wydawnictwa SIGMA-NOT Spółka z o.o., 00-716 Warszawa, ul. Bartycka 20, skr. 1004. Telefony: 40-00-21 wewn. 293, 295, 299 lub 40-30-86, 40-35-89.

Wstępna cena jednego egzemplarza na 1992 rok: normalna – 18 000 zł, ulgowa – 13 500zł

Wartość prenumeraty (w zł):
Normalna: kwartalna – 54 000, półroczna 108 000, roczna 216 000
Ulgowa: kwartalna – 40 500, półroczna 81 000, roczna 162 000

Uwaga: W przypadku zmiany cen w okresie objętym prenumeratą, prenumeratorzy zobowiązani są do dopłaty różnicy cen.

Skład i druk: Drukarnia SIGMA NOT Sp. z o.o. z. 405/92

P.1877/92

MAREK M. STABROWSKI
Instytut Elektrotechniki Teoretycznej
i Miernictwa Elektrycznego
Politechnika Warszawska

AMIL – interakcyjny język symulacji układów dynamicznych

Języki symulacji i analizy układów dynamicznych służą do modelowania różnorodnych układów opisywanych za pomocą równań różniczkowych zwyczajnych i równań algebraicznych. Obecnie swego rodzaju standardem w tej dziedzinie jest język ACSL (*Advanced Continuous System Simulation Language*) [1]. Język ten nie jest zbyt nowoczesnym opracowaniem, gdyż powstał przed 15 laty. Oprogramowanie zintegrowanego otoczenia tego języka jest dostępne tylko w jego najnowszych wersjach. Bardzo istotne jest i to, że cena translatora tego języka jest wysoka i jest on dobrze sprzętowo zabezpieczony przed nie licencjonowanym użytkowaniem. Znacznie nowocześniejsza jest koncepcja języka COSMOS [7], zwłaszcza pod względem naturalności zapisu równań różniczkowych. Kompilator tego języka znajduje się jednak dopiero w stadium testów i obecnie trudno jeszcze mówić o jakiejś jego ustalonej pozycji.

Język AMIL (*Advanced MImic Language*) [11] jest kontynuacją, a zarazem zasadniczym rozwinięciem znanego języka MIMIC [2, 4, 8, 9, 10]. Może on być konkurencyjnym narzędziem symulacyjnym, zwłaszcza jeśli uwzględnić niską cenę i zawartość oprogramowania. Jest to nowoczesny interakcyjny język przeznaczony do użytkowania na komputerach osobistych klasy IBM PC i podobnych. Równania różniczkowe, rozwiązywane za pomocą programów w języku AMIL, mogą być równaniami sztywnymi, gdyż pakiet programów całkowania języka zawiera m.in. programy realizujące dwa warianty metody Geara.

Podobnie jak inne języki tej grupy, język AMIL ma dużą moc opisową, a nauczenie się jego wymaga od początkującego użytkownika zaledwie kilku godzin. Funkcjonalnie język AMIL nie różni się od najnowszych wersji języka ACSL [1], a jest nowocześniejszy składniowo.

Warto wspomnieć o innym bardzo interesującym krajowym opracowaniu – języku PC GODYS (Instytut Maszyn i Sterowania Układów Elektroenergetycznych AGH Kraków) [5], również przeznaczonym dla komputerów osobistych klasy IBM PC. Do zalet tego języka należy możliwość modelowania układów dyskretnych i dość rozbudowana biblioteka makroinstrukcji. Brak w nim, w chwili obecnej, otoczenia w postaci zintegrowanego, zaawansowanego aparatu numerycznego do analizy układów sztywnych oraz eleganckiej grafiki. Kompilator języka PC GODYS jest jednak, podobnie jak AMIL, stale doskonalony i wady te będą stopniowo usuwane.

Podstawowe właściwości użytkowe

Język AMIL jest bardzo łatwy do opanowania. Wydaje się, że dwugodzinna lektura podręcznika języka pozwala pisać poprawne programy rozwiązujące dość złożone zagadnienia, a sześć godzin wystarcza na pełne opanowanie języka.

Charakterystyczną cechą tego języka jest jego paralelizm. Oznacza to, że instrukcje składające się na program w tym języku, poza nielicznymi wyjątkami, mogą być pisane w dowolnej kolejności. Inną zaletą języka jest fakt, że program jest bardzo zbliżony do oryginalnego analitycznego sformułowania problemu. Bardzo istotną jego cechą jest swobodny format programu źródłowego. Ułatwia to pracę w interakcyjnych systemach komputerowych i eliminuje wiele błędów popełnianych przez początkującego użytkownika.

Czynne instrukcje języka składają się z operatorów funkcji standardowych, operatorów arytmetycznych, nazw zmiennych i stałych oraz literałów. Dla funkcji standardowych używa się trzyliterowych nazw mnemonicznych. Za pomocą dwóch przykładów zilustrujemy użycie dwóch podstawowych funkcji (operatorów): całkowania, służącego do rozwiązywania równań różniczkowych oraz funkcji zależności uwikłanej, służącej do rozwiązywania równań algebraicznych.

Operator całkowania ma oznaczenie *INT*, tzn.

$$R = INT(X, XZERO) \tag{1}$$

czyli $R = XZERO + \int X\,dt$, gdzie t jest zmienną niezależną (czas).

Rozwiązywane równania różniczkowe tworzą zazwyczaj pewien układ zawierający zarówno równania pierwszego, jak i wyższych rzędów. Ze względu na dysponowanie w języku AMIL operatorem całkowania, niezbędne jest przekształcenie wszystkich równań wyższego rzędu w układy równań rzędu pierwszego. To proste przekształcenie oraz zastosowanie operatora INT zilustrujemy za pomocą przykładu równania Duffinga [6]. Jest to równanie drugiego rzędu i ma postać:

$$d^2x/dt + 0{,}2\,dx/dt + x^3 = B\cos t \tag{2}$$

W celu przekształcenia tego równania w układ dwóch równań pierwszego rzędu wprowadzamy pomocniczą zmienną y zdefiniowaną jako:

$$y = dx/dt \tag{3}$$

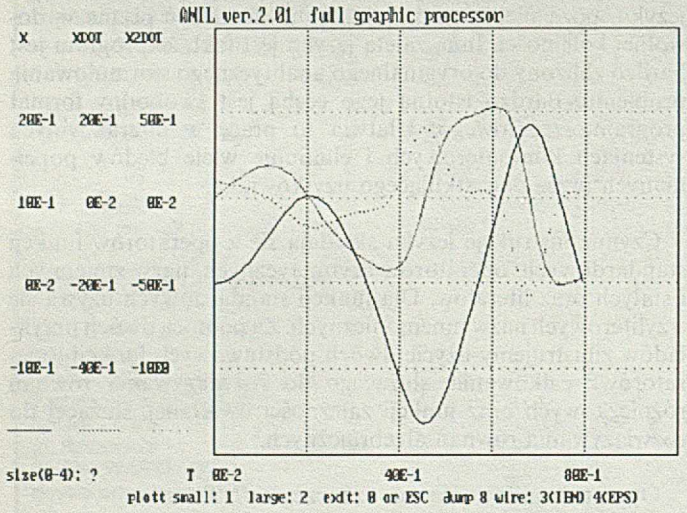
Jest to pierwsze z nowych równań pierwszego rzędu, zastępujących równanie pierwotne (2). Drugie równanie ma postać:

$$dy/dt + 0{,}2\,y + x^3 = B\cos t \tag{4}$$

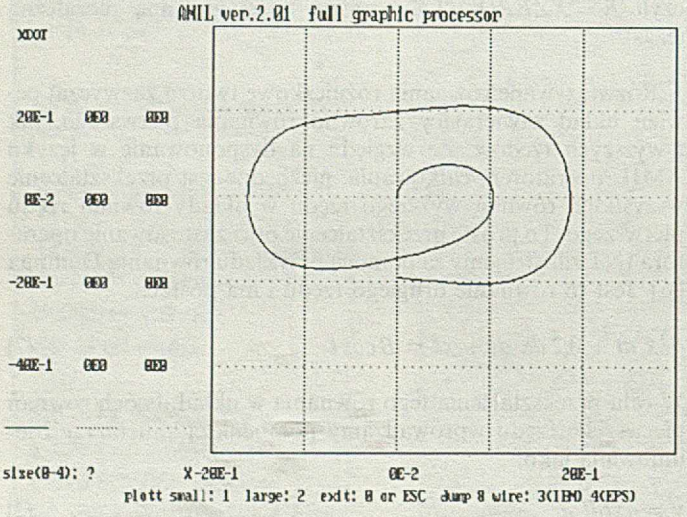
W oparciu na tych dwóch równaniach można napisać program w języku AMIL, mający następującą przykładową postać:

```
RÓWNANIE DUFFINGA
CON(B = 1.0, TSTOP = 8.0, DT = 0.05)
X2DOT =    B*COS(T)-0.2*XDOT-X*X*X
XDOT =     INT(X2DOT, 0.0)
X =        INT(XDOT, 0.0)
           FIN(T, TSTOP)
           PLO(T, X, XDOT, X2DOT)
           PLO(X, XDOT)
           END                          (5)
```

Operator *CON* został użyty do zdefiniowania wartości pewnych stałych (parametrów), w tym m.in. zmiennej specjalnej *DT*, czyli kroku całkowania (komunikacji). Zmienne o nazwach *XDOT* i *X2DOT* to odpowiednio dx/dt i d^2x/dt^2. Operator *FIN* określa warunek końca obliczeń symulacyjnych; w powyższym przykładzie obliczenia należy zakończyć, gdy czas *T* (nazwa zarezerwowana) osiąga wartość parametru *TSTOP*. Funkcje *PLO* służą do generowania wykresów, przy czym w naszym przykładzie poprzestano na automatycznym doborze skali i położenia zera przez kompilator.



Rys. 1. Przebiegi czasowe zmiennych *X*, *XDOT* i *X2DOT* z równania Duffinga (2) wyznaczone za pomocą programu (5) w języku AMIL



Rys. 2. Trajektoria fazowa *XDOT*(*X*) równania Duffinga (2) wyznaczona za pomocą programu (5) w języku AMIL

Rezultaty obliczeń, tzn. dwa wykresy wygenerowane przez funkcje *PLO*, przedstawiono na rysunkach 1 i 2. Rys. 1 przedstawia wykresy zmiennych *X*, *XDOT* i *X2DOT* w funkcji zmiennej *T*, a rys. 2 – trajektorię fazową, tzn. funkcję *XDOT*(*X*).

Występowanie równań algebraicznych (funkcje uwikłane) powinno być w sposób jawny zasygnalizowane w programie za pomocą operatora *IMP*. Rozważmy przykładowe równanie algebraiczne o postaci:

$$x = (10t^2 - x^2)/(3t + 0.01) \tag{6}$$

Odpowiedni fragment programu w języku AMIL, definiujący i rozwiązujący to równanie, ma następującą postać:

```
RIGH1 = 10.*T*T - X*X
X = IMP(X, RIGH1/(3.0*T + 0.01))        (7)
```

W trakcie sortowania programu translator języka AMIL wykrywa i sygnalizuje wszystkie błędy polegające na pominięciu przez autora programu definicji zależności uwikłanych.

Niektóre funkcje specjalne

Jako pierwszą grupę funkcji specjalnych języka AMIL warto omówić funkcje modelujące niektóre często występujące nieliniowości statyczne i dynamiczne. Do grupy tej należą funkcje ograniczenia statycznego *LIM*, ograniczenia dynamicznego *LIN*, integratora rzeczywistego *FTR* oraz strefy martwej *DSP*. Przykładowa funkcja ograniczenia statycznego *LIM* o składni:

$$Y = LIM(X, A, B) \tag{8}$$

nadaje zmiennej *Y* wartość *X*, gdy *X* jest zawarte w przedziale między *A* i *B*. Poza tym przedziałem wartość *Y* równa jest *A*, gdy *X* jest mniejsze od *A* i odpowiednio równa *B*, gdy *X* jest większe od *B*. Strefa martwa jest modelowana np. w sposób następujący:

$$Y = DSP(A, B, C) \tag{9}$$

przy czym dla zmiennej *A*, zawartej w przedziale między *B* i *C*, wartość *Y* jest równa zeru, poniżej *B* wynosi *A*–*B*, a powyżej *C* wynosi *A*–*C*.

Warto jeszcze tu wspomnieć o funkcji *TDL*, modelującej opóźnienia czasowe.

Do grupy funkcji specjalnych języka AMIL można też zaliczyć operatory definiujące funkcje jednej lub dwóch zmiennych za pomocą tabelek (np. wyniki pomiarów rzeczywistego obiektu). Taka definicja funkcji wymaga podania współrzędnych węzłów, niekoniecznie równoodległych, w których jest znana wartość funkcji oraz wartości funkcji w tych węzłach.

Funkcje te definiuje się za pomocą operatorów *CFN* (lub *PFN*) podając nazwę funkcji, np. *FUNX* oraz liczbę zmiennych niezależnych – w tym przykładzie równą 1 – oraz wartości liczbowe jako pary lub trójki liczb dla poszczególnych węzłów, tzn.:

```
FUNX = CFN(1.0/
0.0,9.6, 3.2,6.2, 4.2,6.0,
4.5,6.3, 6.4,8.8)                       (10)
```

Użycie tak zdefiniowanej funkcji tabelarycznej może mieć następującą przykładową postać:

$$VAL = FUN(FUNX, X3) \tag{11}$$

Pierwszym argumentem instrukcji *FUN* powinna być nazwa funkcji występująca w definicji funkcji (w istrukcji z *CFN* lub

PFN), a drugim – wartość współrzędnej (zmienna), dla której interesuje nas wartość funkcji. Rezultatem jest nazwa zmiennej, która ma przyjąć wartość funkcji dla drugiego argumentu. Wyznaczanie wartości funkcji w punktach nie będących węzłami odbywa się przez interpolację liniową lub ekstrapolację.

Warto jeszcze wspomnieć o dwóch specjalnych funkcjach wyprowadzenia na zewnątrz wyników symulacji, wprowadzonych do języka AMIL w związku z konkretnymi potrzebami symulacji złożonych układów pomiarowych [12]; są to instrukcje *OUT* i *OUC*. Instrukcja *OUT* służy do wyprowadzenia do zbioru dyskowego wyników symulacji jako 2-bajtowych liczb całkowitych w kolejnych krokach komunikacji (pozornych krokach całkowania). Instrukcja *OUC* służy do wyprowadzania do zbioru dyskowego nazw stałych, występujących jako argumenty instrukcji *CON/PAR* oraz ich wartości liczbowych jako liczb podwójnej precyzji niesformatowanych. Instrukcja ta jest użyteczna przy opracowywaniu dokumentacji obliczeń symulacyjnych i ich ewentualnym dalszym opracowywaniu.

Funkcje przełączające i logiczne zmienne kontrolne

Funkcje przełączające *LSW* i *FSW* służą do definiowania zmiennych logicznych i numerycznych uzależnionych od czasu. Funkcja *LSW* jest trójargumentową funkcją przełączającą, sterowaną wartością logiczną pierwszego argumentu. Przykładowe użycie tej funkcji ma postać:

$$D = LSW(A, B, C) \tag{12}$$

Jeśli zmienna A ma wartość logiczną „prawda". to rezultat D jest równy wartości argumentu A. W przypadku, gdy zmienna A ma wartość logiczną „fałsz", rezultat D jest równy wartości zmiennej C. Druga z tych funkcji – *FSW* – jest czteroargumentową funkcją przełączającą sterowaną wartością liczbową pierwszego argumentu. Przykładowe użycie tej funkcji ma postać:

$$E = FSW(A, B, C, D) \tag{13}$$

Jeśli zmienna typu numerycznego A ma wartość mniejszą od zera, to rezultat E jest równy zmiennej B. Jeśli $A = 0$, to rezultat E jest równy C. Wreszcie, gdy $A > 0$, to jest $E =$ D. Zastosowanie tej funkcji ilustruje poniższy przykład. Załóżmy, że zmienna $U2$ ma mieć wartość równą 0 dla czasu T mniejszego lub równego 2, a dla dalszych chwil czasowych wartość:

$$U2 = 5.*sin(314.*T) \tag{14}$$

Tego rodzaju funkcję można łatwo zdefiniować za pomocą funkcji przełączającej *FSW*, np.:

$$U2 = FSW(T\text{–}2.,0.,0.,5.*SIN(314.*T)) \tag{15}$$

W języku AMIL ze względu na jego paralelizm jest niezbędne dysponowanie aparatem pozwalającym na omijanie niektórych instrukcji, jeśli jest spełniony określony warunek. Na realizację takich „skoków warunkowych" lub przetwarzania sekwencyjnego, znanego z uniwersalnych języków programowania, pozwalają zmienne sterujące logiczne. Zmiennymi sterującymi logicznymi są wszystkie zmienne występujące w polu rozpoczynającym się w pierwszej kolumnie wiersza. Instrukcje programu opatrzone zmiennymi sterującymi logicznymi są wykonywane wtedy, gdy zmienna sterująca ma wartość „prawda", a są pomijane wtedy, gdy zmienna sterująca ma wartość „fałsz". Rozważmy prosty przykład definiujący pewien przebieg narastający liniowo od chwili $t = 2$, a wcześniej równy zeru, tzn.:

$$\begin{aligned} v &= 0 && \text{dla } t<2. \\ v &= a*(t-2) && \text{dla } t>=2. \end{aligned} \tag{16}$$

Pierwsza metoda definiowania takiego przebiegu nie wykorzystuje zmiennych sterujących logicznych i odpowiednia definicja ma postać:

$$V = FSW(T\text{–}2., 0., 0., A*(T\text{–}2.)) \tag{17}$$

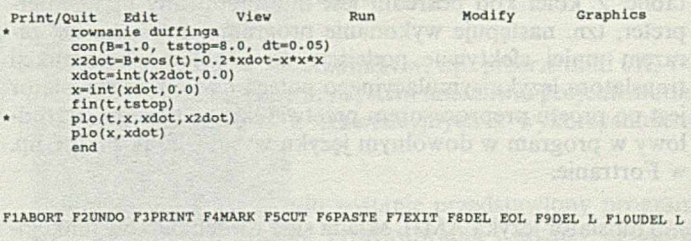
Alternatywna metoda definiowania takiego przebiegu, korzystająca ze zmiennych sterujących logicznych ma następującą postać:

$$\begin{aligned} &TLE2 = FSW(T\text{–}2., TRUE, TRUE, FALSE) \\ &TGT2 = NOT(TLE2) \\ &TLE2 \quad V = 0. \\ &TGT2 \quad V = A*(T\text{–}2.) \end{aligned} \tag{18}$$

W pierwszej metodzie definiowania takiego liniowo narastającego opóźnionego przebiegu, wartości v są obliczane w każdej chwili czasowej i zmienna v przyjmuje tę obliczoną wartość zależnie od znaku wartości wyrażenia t–2. W drugiej metodzie wyrażenie $a*(t\text{–}2)$ jest obliczane tylko wtedy, gdy $t > 2$. Oszczędności obliczeniowe są niewielkie, ale chodziło tu tylko o ilustrację zastosowania zmiennych sterujących logicznych.

Zintegrowane otoczenie kompilatora języka AMIL oraz interakcja

Zintegrowane otoczenie kompilatora języka AMIL obejmuje edytor ekranowy, kompilator, egzekutor, interpreter interakcyjnego oddziaływania na kod wynikowy oraz procesor graficzny. Ze wszystkich tych narzędzi użytkownik może korzystać za pomocą systemu menu (głównego i pomocniczych), podobnego zewnętrznie do menu produktów firmy Microsoft (rys. 3). Użytkowanie zintegrowanego otoczenia języka AMIL nie wymaga więc znajomości systemu operacyjnego.

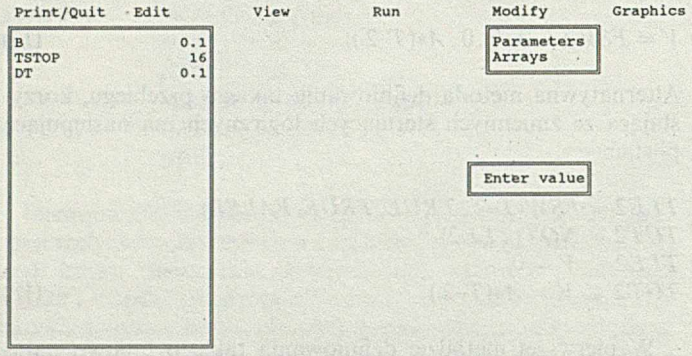


Rys. 3. Menu główne i okno edytora otoczenia zintegrowanego kompilatora języka AMIL

Rozdział kompilacji i wykonania programu w języku AMIL na dwa oddzielne procesy umożliwia interakcyjne oddziaływanie na wartości wybranych zmiennych prostych lub tablic zdefiniowanych jako parametry (operator *CON*), bez powtórnego kompilowania programu. W menu modyfikacji (*MODIFY*) jest prezentowana użytkownikowi lista nazw symbolicznych zmiennych wybranego programu oraz ich wartości liczbowe. Po podjęciu decyzji o modyfikacji wybranej zmiennej, zostaje otwarte okienko dialogowe, w którym użytkownik może podać nową wartość zmiennej. Modyfikacja kodu wynikowego jest procesem natychmiastowym i kod wynikowy może być od razu skierowany do wykonania. Przykładowy wygląd ekranu z menu *MODIFY* dla programu (5) rozwiązującego równanie Duffinga przedstawiono na rys. 4.

Menu postprocesora graficznego (*GRAPHICS*) pozwala generować wykresy o dwóch wielkościach: zajmujących cały ekran monitora oraz 1/4 ekranu. Celem zachowania kompatybilności ze starszymi wersjami kompilatora pozostawiono generowanie

wykresów alfanumerycznych, możliwych do wyprowadzenia na najprostszych drukarkach. W przyszłości przewiduje się rezygnację z tej opcji, odpowiednie modyfikacje składni języka oraz rozbudowę postprocesora graficznego, obecnie nieco skrępowanego współpracą z grafiką alfanumeryczną.



Rys. 4. Menu główne, menu poboczne interakcji *MODIFY* i okienko dialogowe dla programu (5) rozwiązującego równanie Duffinga (2)

Edytor ekranowy jest w swej koncepcji zbliżony do edytora *TED* (Tom Kihlken) i może przetwarzać pliki dyskowe o wielkości nie przekraczającej 64 kB. Operacje wykonywane przez edytor obejmują m.in. przeglądanie plików, oznaczanie bloków, kopiowanie, wycinanie, dołączanie, anulowanie ostatnich poprawek itp.

Ze wszystkich poziomów menu głównego i menu pobocznych jest dostępna kontekstowa informacja („help").

Struktura translatora języka AMIL

Translator języka AMIL wykorzystuje zasadę generowania oraz interpretowania kodu pośredniego. Program źródłowy jest przekształcany w kod pośredni, zapisywany następnie w zespole tablic. Z kolei kod pośredni jest interpretowany przez interpreter, tzn. następuje wykonanie programu. Prostsze, ale zarazem mniej efektywne podejście do problemu konstrukcji translatora języka symulacyjnego polega na tym, że translator jest po prostu preprocesorem przetwarzającym program źródłowy w program w dowolnym języku wysokiego poziomu, np. w Fortranie.

Translator języka AMIL składa się z dwóch bloków funkcjonalnych – kompilatora oraz interpretera-egzekutora. Kod pośredni jest zapisywany w pliku dyskowym i dzięki temu jest możliwe interakcyjne oddziaływanie na zmienne proste i tablice zdefiniowane jako parametry za pomocą operatorów *CON*, *PAR*, *CFN* i *PFN*.

Blok kompilatora generuje kod pośredni zgrupowany w tablicy *INTFRM* oraz tworzy tablice nazw symbolicznych, literałów i stałych tekstowych. Do translacji wyrażeń arytmetycznych wykorzystuje się metodę stosową, przekształcając wyrażenia w postać odwrotną polską *RPN*. W drugim etapie translacji wyrażeń arytmetycznych, postać odwrotną polską przetwarza się w ciąg wyrażeń arytmetycznych elementarnych, najczęściej dwuargumentowych. W tym etapie przetwarzania wyrażeń arytmetycznych translator definiuje nowe zmienne pośrednie. Utworzony w ten sposób kod pośredni nadaje się do przetwarzania za pomocą względnie prostego interpretera.

Ostatnim etapem kompilacji jest sortowanie kodu pośredniego. Kod pośredni jest porządkowany w sposób umożliwiający wykonywanie obliczeń, poczynając od obliczenia wartości wyrażeń arytmetycznych zawierających już znane lub już obliczone wartości zmiennych. Przypomnijmy, iż ze względu na paralelizm języka AMIL jest poprawna następująca sekwencja instrukcji:

$$UCUR = UPEAK*SIN(PHI)$$
$$PHI = OMEGA*T + PHIO \qquad (19)$$

Moduł sortujący porządkuje kod pośredni odpowiadający tym wierszom programu w taki sposób, że następuje efektywna zamiana ich kolejności. Innym nietrywialnym zadaniem modułu sortującego jest wykrywanie nie sygnalizowanych w programie źródłowym pętli algebraicznych. Sortowanie wykorzystuje system trzech atrybutów zmiennych, rozróżniający zmienne jeszcze nie określone, określone oraz określone z użyciem logicznej zmiennej kontrolnej.

Drugim blokiem translatora jest blok interpretera, interpretującego kod pośredni, odczytującego dane oraz dokonującego właściwych obliczeń symulacyjnych. Początkowo rozważano zastosowanie do rozwiązywania układu równań różniczkowych metody Rungego-Kutty o zmiennym kroku całkowania. Jej zaletą jest prostota, zdolność samostartowania oraz zadowalająca na ogół dokładność. Jednakże metoda ta jest nieprzydatna w przypadku układów sztywnych i z tego względu w translatorze języka AMIL zastosowano metody Adamsa i Geara [3, 11]. Stosowanie metody Geara jest zalecane w przypadku sztywnych układów równań różniczkowych, tzn. takich, w których różnice wartości stałych czasowych są duże. Dla każdej z tych metod są dostępne trzy warianty iteracji:

- iterakcja funkcjonalna bez obliczania pochodnych cząstkowych,
- metoda siecznych z jakobianem przybliżanym za pomocą obliczanych wewnętrznie różnic skończonych,
- metoda siecznych z jakobianem zastępowanym przybliżeniem diagonalnym pochodną i kierunkową.

Wybór metody i wariantu iteracji następuje przez nadanie specjalnej zmiennej *METHOD* wartości 1.0...6.0. Wartości *METHOD* z przedziału 1.0..3.0 wybierają metodę Adamsa z jednym z trzech wariantów postępowania iteracyjnego, a wartości z przedziału 4.0...6.0 służą do wyboru jednego z wariantów metody Geara. Jeśli zmienna *METHOD* nie została zainicjowana w programie w sposób jawny, to całkowanie odbywa się przy wykorzystaniu metody Adamsa z przybliżaniem jakobianu za pomocą różnic skończonych. Po zakończeniu obliczeń symulacyjnych jest drukowany wykres zależności czasowych wybranych zmiennych. Moduł programowy drukujący wykresy pozwala na opcjonalne automatyczne dobieranie skali, co jest bardzo wygodne, gdy nie są znane a priori ekstremalne wartości poszczególnych zmiennych. Możliwe jest również drukowanie wykresów trajektorii fazowych przez odpowiedni wybór kolejności zmiennych w instrukcji zlecającej druk wykresu.

* * *

W obecnej wersji (2.10) język AMIL ma na tyle dużą moc opisową, że umożliwia realizację prawie wszystkich eksperymentów symulacyjnych, realizowalnych w języku ACSL. Ze względu na elastyczną i otwartą strukturę translatora, wprowadzenie dodatkowych funkcji standardowych oraz nowych konstrukcji językowych jest względnie łatwe.

Zintegrowane otoczenie języka AMIL może być używane zarówno na sprzęcie z koprocesorem arytmetycznym, jak i na sprzęcie bez takiego koprocesora. Można używać karty graficzne Hercules, CGA, EGA oraz VGA.

dokończenie na s. 8

PIOTR BIZACKI
GRAŻYNA PALIWODA
MAREK SKOMOROWSKI
Instytut Informatyki
Uniwersytet Jagielloński
Kraków

Interakcyjny system do projektowania i analizy cyfrowych układów logicznych

Na wstępie zostaną przedstawione pewne, niezbędne z punktu widzenia użytkownika systemu, wiadomości na temat cyfrowych układów logicznych. Czytelnik zainteresowany teorią układów logicznych może skorzystać z obszernej literatury na ten temat (np. [3, 4, 5, 7]).

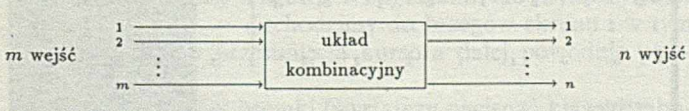
Rozważmy układ logiczny mający m wejść i n wyjść. Załóżmy, że sygnały na każdym wejściu i wyjściu mogą przyjmować jedynie dwie wartości, oznaczone umownie jako 0 i 1. Kombinacja sygnałów wejściowych jest nazywana *stanem wejść*. Podobnie kombinacja sygnałów wyjściowych jest nazywana *stanem wyjść*. Oznaczmy zbiór wszystkich możliwych, dla danego układu, sygnałów wejść przez X, zaś zbiór wszystkich możliwych stanów wyjść przez Y. Działanie układu kombinacyjnego opisuje funkcja logiczna:

$$Y = f(X)$$

równoważna zbiorowi funkcji logicznych

$$y_1 = f_1(x_1, ..., x_m)$$
$$y_2 = f_2(x_1, ..., x_m)$$
$$\vdots$$
$$y_n = f_n(x_1, ..., x_m)$$

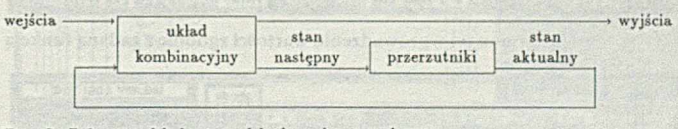
Układ kombinacyjny można również opisać za pomocą tablicy prawdy. Elementarnymi układami logicznymi są bramki logiczne, realizujące proste funkcje logiczne, na przykład *AND*, *OR*, *NOT*, jednej lub wielu zmiennych logicznych. Przedstawiony w pracy system umożliwia projektowanie i analizę układów kombinacyjnych z wykorzystaniem bramek logicznych *AND*, *OR*, *NOT*, *NAND*, *NOR* i *EXCLUSIVE-OR*.



Rys. 1. Schemat blokowy układu kombinacyjnego

Układ logiczny jest rozpatrywany w czasie dyskretnym. Czas układu jest zadany w postaci dyskretnej skali czasu. Kolejne punkty na skali czasu oznaczać będziemy liczbami naturalnymi $1, 2, ..., k, k+1, ...$. Wartości stanów wejść i stanów wyjść w chwili k będziemy oznaczać przez X_k i Y_k. W układzie kombinacyjnym stan wejść X_k wyznacza jednoznacznie stan wyjść Y_k. Charakterystyczną cechą układów kombinacyjnych jest brak pamięci. Schemat blokowy układu kombinacyjnego jest przedstawiony na rys. 1.

W układach sekwencyjnych stan wyjść Y_k zależy nie tylko od stanu wejść X_k, ale także od poprzedniego stanu wyjść Y_{k-1}. Cechą charakterystyczną układów sekwencyjnych jest pamięć. Elementarnym układem sekwencyjnym jest układ zwany przerzutnikiem, którego podstawową funkcją jest pamiętanie jednego bitu informacji. Schemat blokowy układu sekwencyjnego jest przedstawiony na rys. 2.



Rys.2. Schemat blokowy układu sekwencyjnego

Istnieje kilka typów przerzutników, np. przerzutniki *JK*, *D*, *SR*, *T*. Przedstawiony w pracy system umożliwia projektowanie synchronicznych układów sekwencyjnych z wykorzystaniem przerzutników *JK* i *D*.

W dalszej części artykułu zostanie przedstawiony program UL (Układy Logiczne) do projektowania i analizy układów logicznych. Program ten jest wywoływany z poziomu systemu operacyjnego poleceniem UL.

Projektowanie układów kombinacyjnych za pomocą programu UL

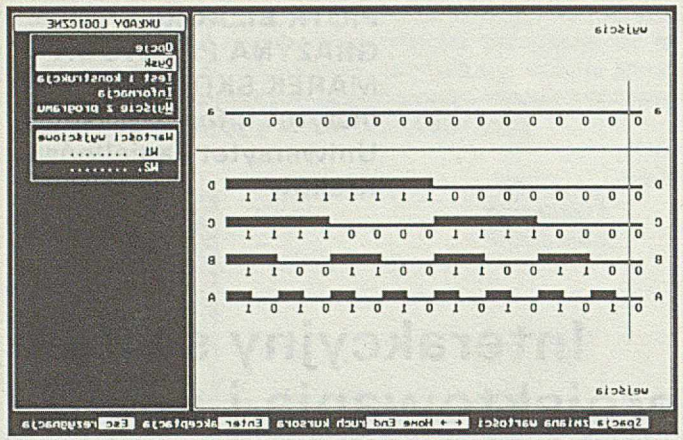
Jako przykład zaprojektujemy układ kombinacyjny, którego działanie opisuje następująca funkcja logiczna:

$$F(D, C, B, A) = \Sigma(0, 1, 2, 4, 5, 6, 8, 9, 12, 13, 14)$$

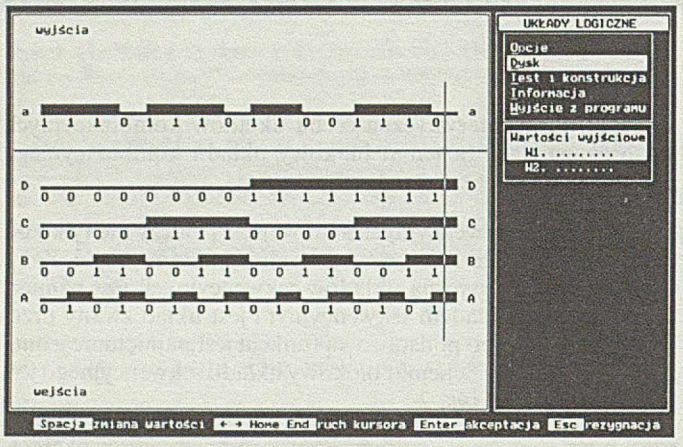
W celu zaprojektowania układu wybieramy polecenie *Dysk/Wartości/Nowe wartości*, uzyskując stan programu jak na rys. 3.

Następnie, za pomocą klawiszy *Spacja*, ←, →, ↑, ↓ ustawiamy wartości wyjścia a (rys. 3) zgodnie z zadaną funkcją, uzyskując stan programu jak na rys. 4.

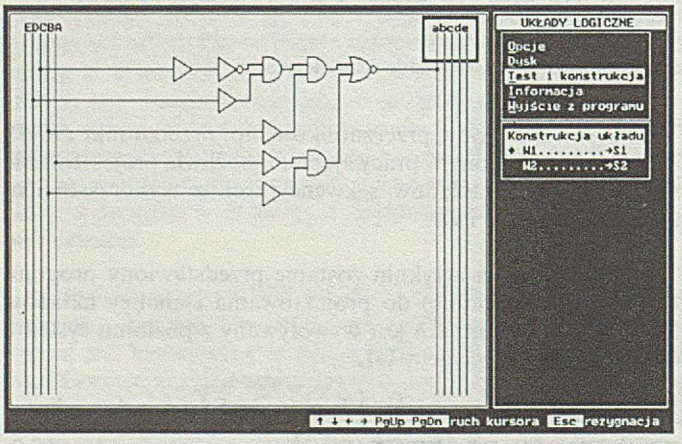
Wciskając klawisz *Enter* akceptujemy ustawienie wartości, a następnie wciskając klawisz *Esc* przechodzimy do głównego



Rys. 3. Stan programu po wyborze polecenia *Dysk/Wartości/Nowe wartości*



Rys. 4. Stan programu po wprowadzeniu wartości zgodnie z zadaną funkcją



Rys. 5. Schemat logiczny układu realizującego zadaną funkcję

menu programu. Wybieramy opcję *Test* i *konstrukcja*, a następnie polecenie *Konstrukcja*. W okienku edycyjnym uzyskujemy schemat logiczny projektowanego układu (rys. 5), wykorzystujący najczęściej spotykane symbole graficzne bramek logicznych (bufor, inwerter, *AND*, *NOR*).

Projektowanie układów sekwencyjnych za pomocą programu UL

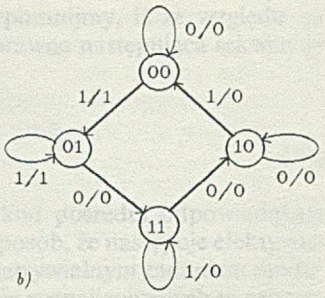
Jako przykład zaprojektujemy układ sekwencyjny, którego graf i tablica stanów są przedstawione na rys. 6.

Z rys. 6 wynika, że projektowany układ będzie miał jedno wejście, jedno wyjście oraz dwa przerzutniki.

a)

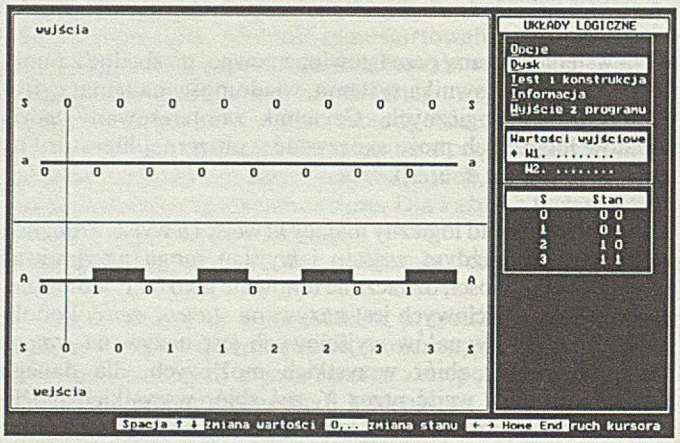
Stan aktualny		Wejście	Stan następny		Wyjście
Q_1	Q_2	A	Q_1	Q_2	a
0	0	0	0	0	0
0	0	1	0	1	1
0	1	0	1	1	0
0	1	1	0	1	1
1	0	0	1	0	0
1	0	1	0	0	0
1	1	0	1	0	0
1	1	1	1	1	0

b)



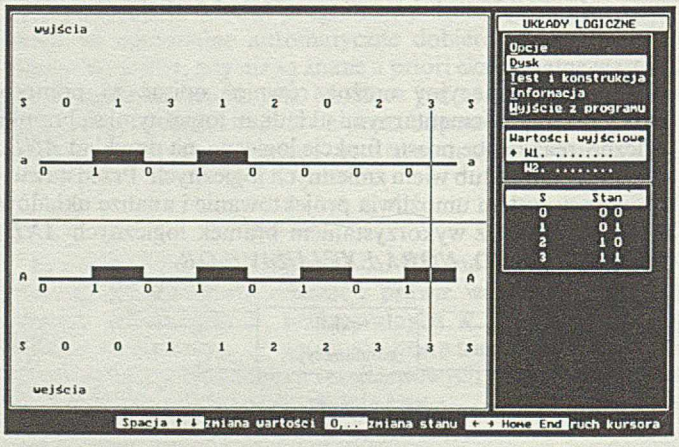
Rys. 6. Tablica (a) i graf (b) stanów projektowanego układu sekwencyjnego

W celu zaprojektowania układu wybieramy polecenie *Dysk/Wartości/Nowe wartości*. Po prawej stronie ekranu pojawią się kolejno okienka z pytaniem o liczbę wejść, wyjść i przerzutników. Za pomocą klawiszy ↑, ↓, ←, →, *Spacja* i *Enter* ustawimy te wartości odpowiednio na 1, 1, 2, uzyskując stan programu jak na rys. 7.



Rys. 7. Stan programu po wprowadzeniu liczby wejść i wyjść przerzutników

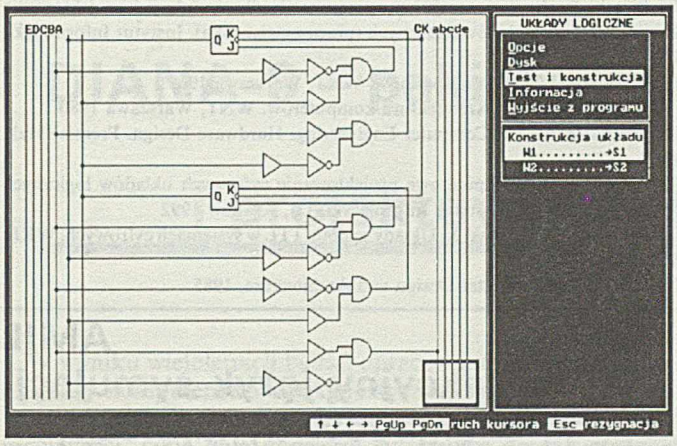
Po prawej stronie ekranu znajduje się tablica z możliwymi stanami przerzutników i odpowiadającymi im liczbami dziesiętnymi. Za pomocą klawiszy *Spacja*, ←, →, *Home*, *End* ustawimy wartości wyjścia, a wprowadzając cyfry od 0 do 3 dokonamy zmian stanu przerzutników, zgodnie z opisem układu na rys. 6. Po tych modyfikacjach uzyskujemy stan programu jak na rys. 8.



Rys. 8. Stan programu po wprowadzeniu opisu działania układu sekwencyjnego, projektowanego na podstawie rys. 6

Wciskając klawisz *Enter* akceptujemy ustawienie wartości, a następnie wciskając klawisz *Esc* przechodzimy do głównego menu programu.. Wybieramy z niego opcję *Test* i *konstrukcja*,

a następnie polecenie *Konstrukcja*. Po prawej stronie pojawi się okienko z pytaniem o typ przerzutników, których chcemy użyć w projektowanym układzie. Za pomocą klawiszy *Spacja* i *Enter* dokonujemy wyboru przerzutników *JK* lub *D*, po czym w okienku edycyjnym uzyskujemy schemat logiczny projektowanego układu (rys. 9).



Rys. 9. Schemat logiczny układu sekwencyjnego, zaprojektowanego z wykorzystaniem przerzutników *JK*

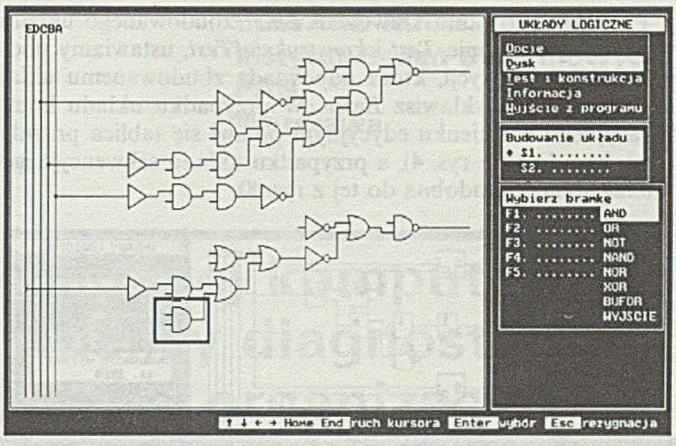
Analiza układów kombinacyjnych i sekwencyjnych za pomocą programu UL

Użytkownik ma do dyspozycji specjalny edytor, wywołany poleceniem *Dysk/Układ/Nowy układ*, umożliwiający budowanie lub modyfikowanie układów, które następnie można poddać analizie, uruchamiając polecenie *Test* i *konstrukcja/Test*.

Po wyborze polecenia *Dysk/Układ/Nowy układ* pojawia się okno z zestawem wszystkich układów. Układy, które nie mają własnej nazwy są przedstawione z nazwą „.....". Za pomocą klawiszy ↑, ↓, *Home*, *End*, a następnie *Enter* wybieramy właściwy układ. Pojawia się jeszcze okienko z pytaniem o liczbę przerzutników w układzie i jeżeli jest ona różna od zera, to również o typ przerzutników. Liczbę przerzutników wybieramy za pomocą klawiszy ↑, ↓, a typ przerzutnika klawiszem *Spacja*. W obu przypadkach naciśnięcie klawisza *Enter* powoduje akceptację wyboru. Przed przejściem do edycji trzeba jeszcze wybrać aktywną bramkę. Pojawia się okienko z bramkami i otwartymi blokami kombinacyjnymi (zbudowanymi i zapamiętanymi układami kombinacyjnymi o jednym wejściu – rys. 10). Klawiszami ↑, ↓, *Home*, *End* dokonujemy wyboru bramki i akceptujemy klawiszem *Enter*. Przechodzimy do edycji schematu.

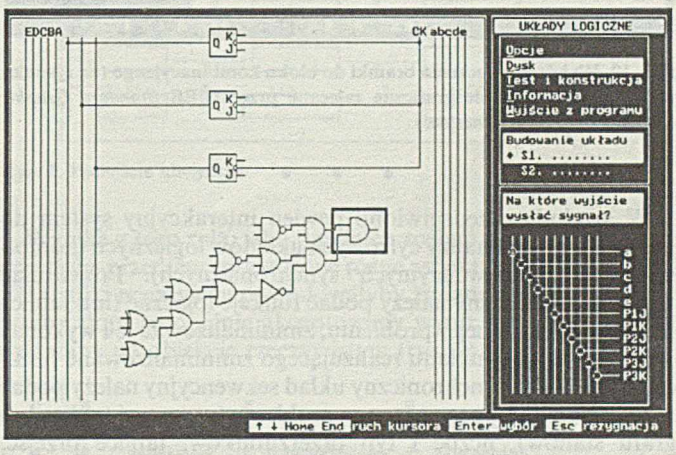
Okno edycyjne ma następujące możliwości:

- Przesuwanie ekranu, tak aby umożliwić edycję całego układu (używamy klawiszy *PageUp* i *PageDown* lub wykorzystując klawisze ↑, ↓, →, ← dochodzimy do brzegów ekranu i w tym momencie próba przesunięcia kursora dalej powoduje przesunięcie ekranu);
- Wybór aktywnej bramki (wystarczy nacisnąć klawisz tabulatora i pojawia się okienko ze wszystkimi bramkami i blokami kombinacyjnymi, jak na rys. 10);
- Tworzenie nowej gałęzi schematu (wystarczy nacisnąć klawisz *Backspace*). Jej korzeniem będzie aktualnie aktywna bramka (lub blok kombinacyjny). Polecenie jest aktywne, jeżeli aktualna bramka jest różna od bramek *WYJŚCIE* i *BUFOR*;
- Przyłączanie nowych gałęzi do wejść przerzutników lub wyjść zewnętrznych (wybieramy *WYJŚCIE* jako aktywną bramkę,



Rys. 10. Wybór aktywnej bramki

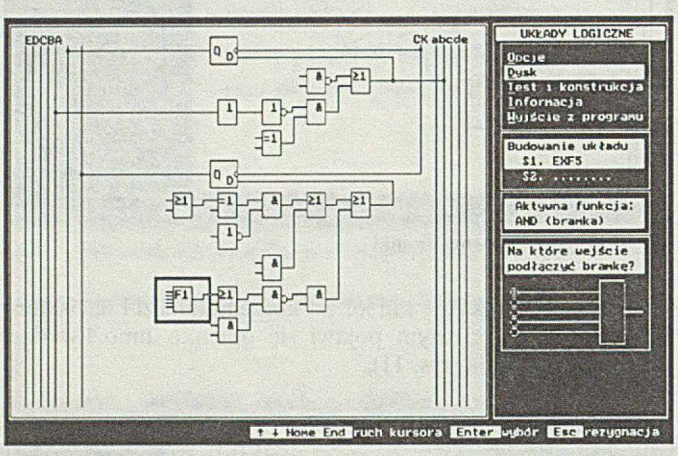
ustawiamy prostokątny kursor na korzeniu gałęzi i naciskamy klawisz *Spacja*, po czym pojawi się okienko umożliwiające wybór przyłączenia, rys. 11);



Rys. 11. Przyłączanie nowych gałęzi schematu

- Przyłączanie bramek do wejść zewnętrznych lub wyjść przerzutników (wybieramy bramkę *BUFOR* i dalej postępujemy podobnie jak w poprzednim punkcie);
- Dołączanie nowych bramek lub bloków kombinacyjnych do układu. Najeżdżamy prostokątnym kursorem na jedną z bramek i naciskamy klawisz *Spacja*. Aktualnie aktywna bramka (lub blok kombinacyjny) zostanie przyłączona na wejście bramki wskazywanej przez kursor. Jeżeli kursor wskazuje blok kombinacyjny, to przed przyłączeniem do niego bramki program zapyta, na które wejście ma przyłączyć bramkę (rys. 12). Pytanie to jest istotne ze względu na to, że funkcja reprezentowana przez ten blok może nie być symetryczna względem swoich wejść, tak jak występuje to w przypadku bramek elementarnych;
- Zamiana bramek (uaktywniamy bramkę, która ma być wstawiona w miejsce innej bramki, ustawiamy kursor na bramce do zamiany i wciskamy klawisz *Insert*);
- Usuwanie niektórych fragmentów układu (ustawiamy kursor na korzeniu fragmentu układu, który chcemy usunąć i wciskamy klawisz *Delete*, po czym program poprosi o potwierdzenie naszej decyzji);
- Zapamiętanie zbudowanego układu i wyjście do menu (klawisz *Enter*). Jeżeli w układzie są gałęzie niepodłączone do wyjść zewnętrznych lub do przerzutników, to program zapyta, czy chcemy je utracić. W przypadku negatywnej odpowiedzi następuje przejście do dalszej edycji schematu;
- Rezygnacja z dokonanych zmian i wyjście do menu (klawisz *Esc*). Program żąda potwierdzenie naszej decyzji.

Po zaakceptowaniu klawiszem *Enter* zbudowanego układu wybieramy polecenie *Test* i *konstrukcja/Test*, ustawiamy podświetlenie na pozycji, która odpowiada zbudowanemu układowi i wciskamy klawisz *Enter*. W przypadku układu kombinacyjnego w okienku edycyjnym pojawi się tablica prawdy (podobna do tej z rys. 4), a przypadku układu sekwencyjnego tablica stanów (podobna do tej z rys. 8).



Rys. 12. Wybór przyłączenia bramki do bloku kombinacyjnego (na rysunku wykorzystano symbole graficzne zalecane przez IEEE *Standard Graphic Symbols for Logic Functions*)

* * *

W artykule przedstawiono pewien interakcyjny system do projektowania i analizy cyfrowych układów logicznych (kombinacyjnych i sekwencyjnych synchronicznych). Projektując układ kombinacyjny należy podać funkcje logiczne (lub tablice prawdy) rozważanego problemu, zminimalizować je i wykonać schemat logiczny układu realizującego zminimalizowane funkcje. Projektując synchroniczny układ sekwencyjny należy podać sekwencję stanów rozważanego problemu (w postaci tablicy lub grafu stanów), liczbę i typ przerzutników, tablice przejść, zminimalizować funkcje logiczne generujące wejścia przerzutników (funkcje wejść przerzutników) oraz wykonać schemat logiczny układu realizującego zminimalizowane funkcje.

Projektując układy kombinacyjne za pomocą programu UL należy podać funkcje logiczne lub tablice prawdy. Projektując układy sekwencyjne należy podać tablicę lub graf stanów, liczbę i typ przerzutników (*JK* lub *D*). Program UL umożliwia w prosty sposób wczytywanie danych, opisujących działanie układów kombinacyjnych (rys. 4) i sekwencyjnych (rys. 8). Do minimalizacji funkcji logicznych użyto metody Quine'a-McCluskey'a. Generowane schematy logiczne układów, mogą wykorzystywać zarówno najczęściej używane jak również zalecane [4] przez IEEE symbole graficzne (rys. 12). Program UL umożliwia również wczytywanie schematów logicznych układów oraz ich analizę.

Oprócz wymienionych wyżej, program UL ma możliwość zapisu na dysk funkcji logicznych, tablic stanów, bloków kombinacyjnych (np. blok kombinacyjny *F*1 na rys.12) oraz schematów logicznych układów. Dane te mogą być następnie odczytywane i modyfikowane. Na otwarte wejścia bramek (tzn. wejścia nigdzie nie przyłączone) użytkownik może podać zero lub jedynkę logiczną. Możliwe jest również uzyskanie informacji o liczbie bramek projektowanego lub analizowanego układu. Użytkownik może ustalić katalog, na którym dane będą zapisywane na dysk. Dokładny opis programu UL podano w [1] i [5].

Program UL został napisany w języku Pascal. W aktualnej wersji ma on pewne ograniczenia na maksymalną liczbę bramek w jednym układzie (150), wejść i wyjść (5), bloków funkcyjnych (5) oraz liczbę przerzutników (3). Wynikają one z ograniczeń pamięciowych sprzętu, na ktrym program został zaimplementowany.

LITERATURA

[1] Bizacki P.: Implementacja oprogramowania dla symulacji i projektowania kombinacyjnych układów logicznych (praca magisterska). Instytut Informatyki, UJ, Kraków 1992
[2] Bromirski J.: Teoria automatów. WNT, Warszawa 1969
[3] Morris Mano M.: Architektura komputerów. WNT, Warszawa 1980
[4] Morris Mano M.: Computer Engineering: Hardware Design. Prentice Hall, 1988
[5] Paliwoda G.: Komputerowe projektowanie cyfrowych układów logicznych (praca magisterska). Instytut Informatyki, UJ, Kraków 1992
[6] Pieńkos J., Turczyński J.: Układy scalone TTL w systemach cyfrowych. WKŁ, Warszawa 1980
[7] Shiva S.G.: Computer Design and Architecture, 1985.

AMIL interakcyjny język symulacji układów dynamicznych

dokończenie ze s. 4

Programy zintegrowanego otoczenia zajmują w obecnej wersji ok. 400 kbajtów, ale technika zbliżona do nakładkowania ogranicza zapotrzebowanie na pamięć operacyjną do ok. 220 kbajtów. Zintegrowane otoczenie jest dostosowane do wykorzystania w sieciach.

Przewidywane modyfikacje języka AMIL, kompilatora oraz zintegrowanego otoczenia, obejmują wprowadzenie mechanizmu makroinstrukcji wymienione już modyfikacje postprocesora graficznego, modelowanie elementów dyskretnych, wprowadzenie naturalnego zapisu równań różniczkowych [7] i wreszcie przeniesienie całości oprogramowania w środowisko systemu operacyjnego UNIX. Powodzenie tych projektów zależy od uzyskania jakiegokolwiek wsparcia sprzętowego i finansowego.

LITERATURA

[1] Advanced Continuous System Simulation Language (ACSL) Reference Manual. Mitchell and Gauthier Associates, Concord Mass. 1986 (4-th ed.)
[2] Budziaszek J.: Język symulacyjny MIMIC. Wydawnictwa AGH, Kraków 1979
[3] Byrne G.D., Hindmarsh A.C.: A Polyalgorithm for the Numerical Solution of ordinary Differential Equations. ACM Transactions on Mathematical Software, 1975, vol. 1, pp. 71–96
[4] CDC-6000 Computer Systems MIMIC Digital Simulation Language Reference Manual. CDC Graphic Division, Minneapolis 1972
[5] Gajda J.: Analiza i optymalizacja mierzalności modeli obiektów identyfikacji. Materiały Sympozjum „Modelowanie i Symulacja Systemów Pomiarowych", Wyd. AGH, Kraków 1991, s. 106–114
[6] Hayashi Ch: Drgania nieliniowe w układach fizycznych. Warszawa, WNT 1969
[7] Kettenis D.L.: COSMOS: A simulation language for continuous, discrete and combined models. Simulation 1992, Vol. 58, No. 1, pp. 32–41
[8] Sansom F.J., Peterson H.E.: MIMIC Programming Manual. Technical Report SEG-TR-67-31, Wright-Patterson AFB, 1967
[9] Stabrowski M.: Translator języka symulacji układów dynamicznych MIMIC-400. Prace IV Konferencji Użytkowników Minikomputera Mera-400, s. 58–63, Gdańsk, 1986
[10] Stabrowski M.: Język symulacji układów dynamicznych MIMIC-400. Aparatura Naukowa i Dydaktyczna, nr 6, 1985, s. 3–6
[11] Stabrowski M.: Język symulacji układów dynamicznych AMIL i jego otoczenia zintegrowane. Materiały Sympozjum „Modelowanie i Symulacja Systemów Pomiarowych", Wyd. AGH, Kraków 1991, s. 72–78
[12] Wierciak J.: Symulacja stanowiska do badania statycznych właściwości mikrosilników prądu stałego. Politechnika Warszawska, Instytut Konstrukcji Przyrządów Precyzyjnych i Optycznych; Sprawozdanie z pracy naukowo badawczej nr 503/113/053, Warszawa 1991.

EDWARD MICHALEWSKI
Instytut Badań Systemowych
PAN
Warszawa

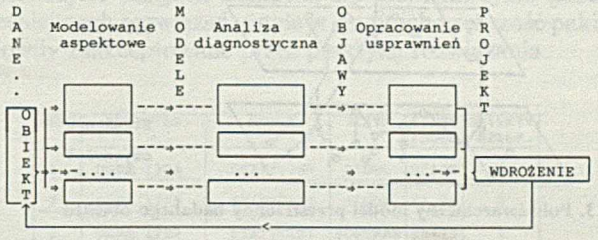
DIANA-9 – pakiet wspomaganej komputerowo analizy diagnostycznej i projektowania struktur organizacyjnych

W wyniku wieloletnich badań opracowano metodykę wspomaganej komputerowo DIAgnostycznej ANAlizy i projektowania systemów zarządzania (DIANA). Wykorzystano w niej możliwości, jakie daje podejście systemowe do tak złożonej problematyki.

Metodyka ta pozwala przeprowadzić własnymi siłami wszechstronną analizę diagnostyczną systemu zarządzania, dokonać zmian usprawniających oraz zaprojektować nową strukturę organizacyjną wraz ze sprawdzeniem efektywności wprowadzonych zmian na modelu symulacyjnym. Dopiero po tym etapie można przystąpić do projektowania systemu informatycznego. Wychodzimy bowiem z założenia iż nawet najlepszy system informatyczny przy „chorym zarządzaniu" nie wiele może pomóc. Odpowiednie procedry umożliwiają wytypowanie czynności nadających się do informatyzacji. Dla nich możemy wprowadzić algorytmy, zaś odpowiednie procedury wygenerują je w kodzie wynikowym. Do przeprowadzenia całego cyklu tych badań służy pakiet programów, którego najnowszą wersję (9.) opracowano dla komputerów klasy IBM PC AT. Ma on wszystkie cechy narzędzi typu CASE-tool.

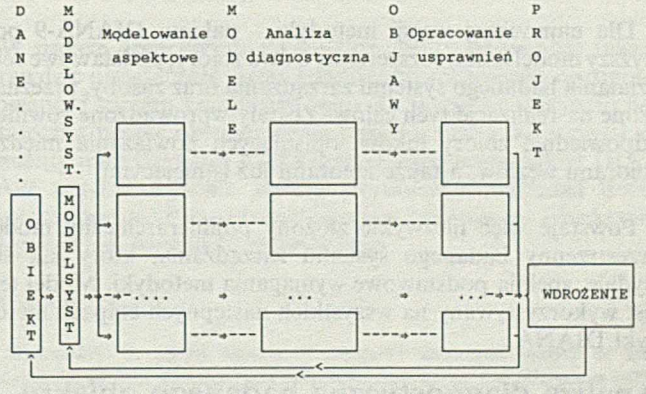
W klasycznym podejściu [8] badany obiekt jest rozpatrywany wieloaspektowo w sposób wzajemnie niezależny (rys. 1).

Uwzględniając sprzeczne nieraz interesy może się zdarzyć, że wynik końcowy dla badanego obiektu będzie nader niekorzystny.



Rys. 1. Podejście klasyczne

Wynika z tego rola, jaką powinien spełnić model, na którym zostałyby przedtem sprawdzone różne aspekty wielu rozwiązań; ich wyniki negatywne powodowałyby zmianę założeń aspektowych i dopiero ogólnie akceptowany wynik pozytywny byłby wdrażany na obiekcie rzeczywistym (rys. 2).



Rys. 2. Podejście systemowe

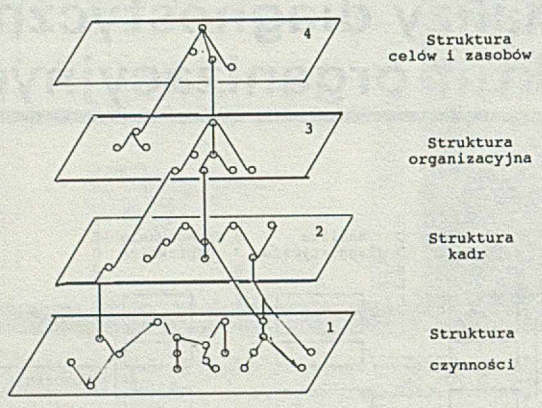
Widać również wymagania, jakie musiałby spełniać taki model. Powinen on w zasadzie, uwzględniać wszystkie możliwe aspekty badań obiektu rzeczywistego (stąd jego nazwa: model systemowy). Byłby to oczywiście model idealny. Opracowaniu takiego modelu dla systemu zarządzania poświęcono sporo czasu przy tworzeniu kolejnych wersji metodyki DIANA. Obecna, dziewiąta wersja, w dużym stopniu spełnia te założenia.

Metodyka DIANA

Już w pierwszej wersji metodyki DIANA przyjęto założenie, że model systemu zarządzania badanego obiektu stanowi sieć powiązań informacyjnych [1]. W sieci tej węzłami są elementarne czynności (operacje) wykonywane przez personel badanego obiektu, zaś łukami – wzajemnie przekazywane informacje, będące wynikami realizacji tych czynności.

Dr inż. EDWARD MICHALEWSKI adiunkt Instytutu Badań Systemowych PAN w Warszawie, ukończył w 1964 roku Wydział Automatyki i Przyrządów Precyzyjnych Politechniki Kijowskiej. W 1968 roku obronił pracę doktorską w dziedzinie teorii sterowania optymalnego i rozpoczął pracę w Instytucie Automatyki PAN (obecnie IBS PAN). Od 1969 roku kieruje zespołem prowadzącym badania w zakresie metod algorytmizacji złożonych procesów. W wyniku prac tego zespołu powstała metodyka wspomaganej komputerowo DIAgnostycznej ANAlizy i projektowania systemów zarządzania (DIANA) oraz kolejne jej aplikacje: pakiety DIANA-1 – DIANA-7 dla dużych komputerów i DIANA-8, DIANA-9 dla techniki mikrokomputerowej. W 1992 r. został kierownikiem Ośrodka Doradztwa Naukowego i Szkolenia IBS PAN. Opracował ponad 100 publikacji naukowych.

Pierwotny model – sieć powiązań między czynnościami, stanowi najniższy poziom obecnego modelu (rys. 3). Może ona mieć strukturę hierarchiczną (zadania, podzadania, operacje). Istnieje również następny, wyższy poziom – sieć powiązań między pracownikami wykonującymi te czynności. Powiązania te (poziome) wynikają z powiązań informacyjnych między czynnościami (czyli z dolnego poziomu). Istnieją również powiązania pionowe między tymi poziomami: między pracownikami i wykonywanymmi przez nich czynnościami [2]. Ten poziom ma też własną hierarchię: od dyrektora naczelnego poczynając, a na szeregowym pracowniku kończąc.



Rys. 3. Polihierarchiczny model przestrzenny badanego obiektu

Wreszcie ponad tym istnieje poziom komórek organizacyjnych (mający oczywiście własną hierarchię, np. poczynając od pionów a na stanowiskach kończąc). Powiązania między komórkami wynikają z powiązań między pracownikami zatrudnionymi w tych komórkach (powiązania poziome). Implikuje to w sposób oczywisty powiązania pionowe między tym poziomem i pozostałymi dwoma poziomami.

Dla najnowszej wersji metodyki – pakietu DIANA-9 powyższy model został rozszerzony o dwa zbiory; podstawowe cele działania badanego systemu zarządzania oraz zasoby, przeznaczone na realizację tych celów. Zostały wprowadzone również odpowiednie zbiory łuków, opisujących powiązania między zbiorami węzłów, a także zbiorami już istniejącymi.

Powstaje więc niezwykle złożony polihierarchiczny model przestrzenny badanego systemu zarządzania, który jak się wydaje, spełnia podstawowe wymagania metodyki. Model ten jest wykorzystywany na wszystkich następnych etapach metodyki DIANA..

Analiza diagnostyczna badanego obiektu

Wykorzystując parametry opisujące zarówno węzły, jak i łuki sieci, przeprowadza się kompleksową analizę diagnostyczną badanego obiektu, której wyniki są wykorzystywane do opracowania kolejnych wersji usprawnień – ponownie modelowanych i diagnozowanych, aż do uzyskania zadowalającego projektu. Dopiero tak usprawniony na modelu obiekt stanowi podstawę do projektowania.

Blok wspomaganej komputerowo analizy diagnostycznej pakietu DIANA-9 zawiera 62 algorytmy (docelowo – 64) wykrywające różne nieprawidłowości na poszczególnych poziomach modelu, jak np.:

- Poziom funkcjonalny:
  - brak rzeczywistego odbiorcy,
  - brak synchronizacji w czasie,
  - dysfunkcjonalność,
  - wadliwe funkcjonowanie kontroli.
- Poziom kadr:
  - nadmierna rozbieżność oceny własnej i przełożonego,
  - nadmierna rozbieżność predyspozycji i wykonywanych funkcji,
  - brak satysfakcji z wykonywanej pracy,
  - nieodpowiednie kwalifikacje zawodowe.
- Poziom komórek organizacyjnych
  - rozbieżność hierarchii stanowisk,
  - nierównomierne obciążenie komórek,
  - nieodpowiedni podział na komórki,
  - nieodpowiedni rozkład sfer działalności.
- Poziom celów i zasobów:
  - niewłaściwe cele dla komórki organizacyjnej.
  - niewłaściwa realizacja celów stanowiska przez zadania.
  - nieodpowiednie zasoby dla realizacji celów komórek.
  - rozbieżność zasobów niezbędnych i faktycznych.

Wykryte objawy są przekazywane do macierzy diagnostycznej, która realizuje diagnozę, tzn. określa zespół przyczyn wywołujących te objawy. Ułatwia to umiejscowienie źródeł niedomagań, określenie ich rodzaju („uczulając" odpowiednich specjalistów) oraz stosowanie środków przewidzianych w pakiecie DIANA-9 do ich usuwania.

Warto zwrócić uwagę, że w zestawie algorytmów wykrywających niedomagania oraz w macierzy diagnostycznej została zawarta cała nagromadzona wiedza autorów metodyki, oparta zarówno na własnym doświadczeniu, jak też na wynikach opublikowanych w literaturze światowej. Pod tym względem pakiet DIANA-9 ma cechy systemu ekspertowego.

Wyniki diagnozy stanowią podstawę do opracowania projektów usprawnień organizacyjnych. Projekty usprawnień organizacyjnych są kolejno sprawdzane na modelu. Dopiero najlepszy wariant, mający zarówno najmniejszą intensywność wykrytych niedomagań, jak też uznanie użytkowników (a więc uwzględniający ocenę czynników nie poddających się formalizacji) stanowi podstawę dalszych prac, lub – w przypadku ich zaniechania – może być wdrażany na obiekcie rzeczywistym.

Projektowanie struktur organizacyjnych

Blok wspomaganego komputerowo projektowania pakietu DIANA-9 również operuje na opisanej wyżej przestrzenno-hierarchicznej sieci. Do projektowania wykorzystuje się tzw.. zalążki – najbardziej istotne dla projektowanych komórek organizacyjnych stanowiska, wyznaczane przez zespół projektantów (najbardziej doświadczonych pracowników badanego obiektu).. Komputer, realizując algorytm typu *cluster-analysis*, ściąga do tych stanowisk najsilniej powiązane z nimi inne stanowiska. Miarą jakości projektowanych komórek jest tzw. siła powiązań, która świadczy o zawartości czynności wykonywanych wewnątrz komórek, zaś jakość całego projektu określa tzw. miara rozproszenia, charakteryzująca powiązania między komórkami. W trakcie projektowania dążymy do maksymalizacji siły powiązań i minimalizacji miary rozproszenia.. Mówiąc obrazowo, dążymy do zachowania zasady „zamkniętych drzwi" wg której urzędnicy większość spraw załatwiają w swoich pokojach (komórkach organizacyjnych) i tylko zakończone zadania przekazują do innego pokoju. Realizacja tego procesu dla kolejnych poziomów struktury daje możliwość uzyskania kompletnego projektu organizacyjnego badanego systemu zarządzania [3].

Również i tu istnieje możliwość sprawdzenia, najpierw na modelu [4], wielu wariantów projektu organizacyjnego. Uzyskany projekt końcowy stanowi podstawę do prac w zakresie komputeryzacji [5], lub – w przypadku rezygnacji z dalszych

etapów – może być podstawą wdrożenia nowej struktury organizacyjnej badanego obiektu.

Opracowanie projektu komputeryzacji obiektu

W pierwszym kroku tego etapu wyodrębnienia się tzw. dendryty zadań [6]: dla wybranego zadania (jest to dla komputera zadanie startowe) można z całej sieci odtworzyć pełną drogę jego realizacji – w postaci drzewa: zadanie startowe – jego dostawcy – dostawcy jego dostawców itd. Jeżeli założyć, że zestaw zadań startowych będzie stanowił wyjście przyszłego systemu informatycznego, to kolejne dendryty tych zadań utworzą podsieć tego systemu, wyodrębnioną z całej sieci badanego obiektu (obszaru).

W następnym kroku wytypowane do informatyzacji zadania rozbijamy na podzadania, te zaś na ich podzadania itd., aż do uzyskania tzw. elementarnych operacji, czyli takich czynności dla których jesteśmy w stanie opisać formalny algorytm ich realizacji. Odpowiednie procedury umożliwiają wprowadzenie tych algorytmów do komputera.

Ostatnim krokiem jest opracowanie wejściowych i wyjściowych ekranów i sprawdzenie uzyskanego projektu komputeryzacji obiektu na modelu za pomocą interpretera. Po uzyskaniu zadowalających wyników możemy otrzymany projekt wygenerować w kodzie źródłowym, w języku C.

Można więc stwierdzić, że obecna wersja pakietu ma wszystkie elementy pakietów klasy CASE-tool.

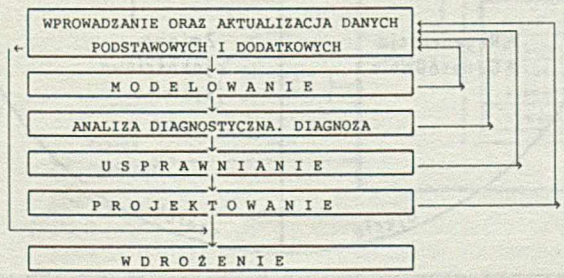
Dalsze postępowanie zależy od stopnia informatyzacji obiektu. W przypadku „dziewiczym" możemy od razu skomplikować projekt jako program użytkowy. Następne jego modyfikacje (m.in.. przez dołączanie kolejnych algorytmów) zgodnie z podaną wyżej metodyką dają gwarancję uzyskiwania nowych wersji nie sprzecznych z poprzednimi. Dla obiektów zaawansowanych informatycznie uzyskujemy kompletny projekt komputeryzacji nowych obszarów jego działalności.

Na zakończenie warto jeszcze zwrócić uwagę na to, że metodyka DIANA umożliwia symulację na modelu również nie istniejącej (nowej) części badanego obiektu, co może stanowić etap pośredni [7] do właściwego projektowania.

Pakiet DIANA-9

Pakiet DIANA-9 jest bradzo złożonym produktem programowym, składającym się z kilkudziesięciu programów (około 100), stanowiący własny oryginalny dorobek. Realizuje on w pełnym zakresie przedstawioną metodykę DIANA (rys. 4).

Pakiet oprogramowano w języku C, w stylu „przyjaznym dla użytkownika", dzięki czemu od użytkownika nie wymaga się przygotowania informatycznego.



Rys. 4. Schemat funkcjonalny pakietu DIANA

Pakiet wykorzystuje oryginalny, opracowany w Instytucie Badań Systemowych PAN, system zarządzania nowoczesną, tzw. postrelacyjną bazą danych o nazwie MIDAS. Dzięki takiemu rozwiązaniu jest możliwe działanie na dużych zbiorach sieciowych w czasie dopuszczalnym dla techniki mikrokomputerowej oraz przyjęcie w tej wersji metodyki trybu konwersacyjnego.

Warto podkreślić, że od pierwszej wersji realizacji metodyki DIANA (rok 1970, pakiet DIANA-1, dla komputera ODRA 1204) przyjęto zasadę, że musi ona przejść testowanie na obiekcie rzeczywistym. Tylko te rozwiązania, które znajdą praktyczne zastosowanie mogą wejść do następnej wersji. W tabeli zawierającej zestawienie kolejnych wersji pakietu DIANA, ostatnia kolumna podaje liczbę przebadanych obiektów, natomiast przedostatnia odzwierciedla procentowy udział wcześniejszej wersji w następnej – właśnie wg zweryfikowanej praktycznej przydatności. Warto zwrócić uwagę, że o ile w pierwszych wersjach zachował się niewielki tylko procent wcześniejszych rozwiązań, o tyle w ostatnich większość pakietu stanowiły zaakceptowane przez praktykę rozwiązania.

WERS	KOMPUTER	MODEL	JĘZYK	STARE /NOWE	OBIEKTY RZECZ.
1	ODRA 1204	STATYCZNY	ALGOL	0/100	1
2	ODRA 1204	STATYCZNY	ALGOL	5/ 95	2
3	ODRA 1204	STATYCZNY	FORTRAN	25/75	4
4	IBM360/R20	STATYCZNY	COBOL	45/55	4
5	ODRA 1325	DYNAMICZNY	FORTRAN	55/45	6
6	ODRA 1305	DYNAMICZNY	FOR/COB	65/35	7
7	HONEYWELL 64	DYNAMICZNY	PLAN	75/25	9
8	IBM PC XT/AT	POLIHIERARCH	C	70/30	3
9	PC AT,PS/2	POLIHIERARCH	C	80/20	5

Porównanie kolejnych wersji pakietu DIANA

W trakcie opracowywania kolejnej wersji starano się także sprawdzić przydatność coraz bardziej nowoczesnego oprogramowania i sprzętu komputerowego. To wszystko wpłynęło na ostateczną postać obecnej wersji pakietu, która ma wszystkie elementy narzędzi klasy CASE-tool. Dzięki temu pakiet DIANA-9 uzyskał wysokie walory użytkowe i wzbudza duże zainteresowanie. Prowadzone obecnie badania na obiektach rzeczywistych w pełni to potwierdzają.

LITERATURA

[1] Michalewski E. „Some aspects of computer diagnostic analysis of the management systems. Control and Cybernetics, Vol. 4, No. 3–4, Warszawa 1975
[2] Michalewski E.: Polyhierarchical dynamic model of a large – scale management system. Prace Naukowe ICT PW, nr 3, Wrocław 1978
[3] Michalewski E.: Methodology of design of information structure based on the use of microcomputer. I Polish Scandinavian Seminar „Current trends in information systems development methodologies", Gdańsk 1988
[4] Michalewski E..: Wskaźniki rozmyte przy projektowaniu dużych sieci. VI Ogólnopolskie Sympozjum SPD-6 „Symulacja procesów dynamicznych", Zakopane 1990
[5] Michalewski E.: Wykorzystanie techniki mikrokomputerowej do projektowania systemów informatycznych. Międzynarodowa konferencja „Nowoczesne metody zarządzania", Wrocław 1990
[6] Michalewski E.: Zastosowanie pakietu DIANA-9 jako CASE. Konferencja „Badania Operacyjne i Systemowe", BOS'91, Warszawa 1991
[7] Michalewski E.: Nowy trend w CAMS – komputerowy lekarz systemu zarządzania. INFOGRYF'90, Szczecin 1990
[8] Zieleniewski J.: Organizacja zespołów ludzkich. PWN, Warszawa 1979.

Już najwyższa pora zaprenumerować INFORMATYKĘ

AGNIESZKA SZEWCZYK
Instytut Cybernetyki Ekonomicznej
i Informatyki
Uniwersytet Szczeciński

Diagnostyka systemów informacyjnych (1)

Problemy podstawowe

Współczesny system informacyjny powinien aktywnie wspomagać człowieka przy podejmowaniu różnorodnych decyzji, zwłaszcza w sferze gospodarowania. Funkcję tę może spełniać taki system, którego charakterystyki jakościowe odpowiadają zmiennym warunkom otoczenia [4].Dla przykładu, system informacyjny przedsiębiorstw, w miarę wzrostu ich samodzielności działania w warunkach rynkowych musi dostarczać nowej jakościowo informacji dotyczącej nie tylko o aktualnej kondycji przedsiębiorstwa, ale również o warunkach wewnętrznych i zewnętrznych działania, ich zmianach, prognozach i przewidywanych skutkach bieżąco podejmowanych decyzji.

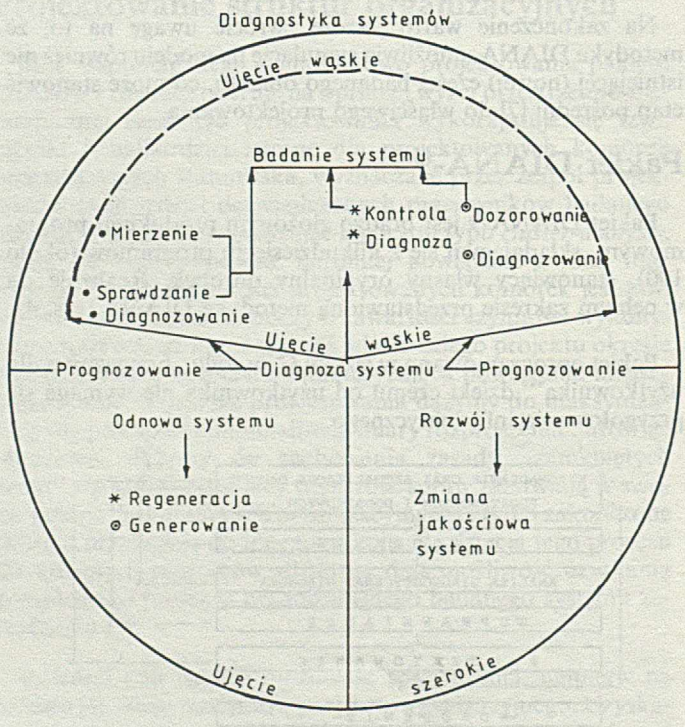
W przypadku systemów informacji naukowo-techniczno-ekonomicznej rosną wymagania dotyczące głównie dyferencyjności, czyli zdolności dzielenia wg określonych kryteriów otrzymanej informacji, jej relewantności i selektywności, tj. zdolności dostosowywania pod kątem potrzeb użytkowników. Rozwój nauki sprawia bowiem, że z jednej strony rośnie lawinowo liczba fachowych publikacji z wielu dziedzin, a z drugiej – badania naukowe, prowadzone bez aktualnej znajomości dotychczasowych dokonań, są w dużej mierze narażone na niebezpieczeństwo plagiatu, a przede wszystkim bezcelowego działania.

W przypadku systemów wyszukiwania informacji wymagania obejmują głównie uzyskanie informacji kompletnej, zwięzłej, komunikatywnej i ergonomicznej, czyli przyswajalnej bez strat czasu i energii użytkowników, dostarczonej w krótkim czasie. Konieczne bywa również umożliwienie rozszerzalności zakresu informacji oraz jej dyspozycyjności, tzn. zdolności do rozporządzania nią wg uznania użytkownika, umożliwiającej wykorzystywanie informacji mimo defektów sprzętu. Cechy te dają bowiem, zwłaszcza w tego typu systemach, wymierne korzyści ekonomiczne. Krótki czas reakcji systemu ma także duże znaczenie psychologiczne i propagandowe, wywiera bowiem wrażenie na użytkowniku i podnosi zaufanie do systemu jako konsultanta. Istotne wydają się również takie cechy konstruujące systemy tej klasy, jak: otwartość, autonomiczność, probabilizm działania, wysoka złożoność itp.

Celowość diagnozowania systemów informacyjnych

Już zasygnalizowanie zagadnień wzrastających funkcji systemu informacyjnego na przykładzie tych trzeba typów systemów informacyjnych, wskazuje nie tylko na różnorodność problemów, ale i na drugi nurt zagadnień, dający się ująć jednym, zasadniczym pytaniem: czy aktualny stan funkcjonujących systemów informacyjnych może sprostać tym wzrastającym wymaganiom? Odpowiedź na to pytanie może dać przeprowadzenie procedury diagnostycznej systemu informacyjnego oraz określenie stanu systemu.

W aspekcie powyższych ustaleń przyjmuje się, że diagnostyka to proces obejmujący formułowanie diagnozy warunkującej odnowę systemu. Różne ujęcie zakresu diagnostyki obrazuje rysunek 1.



Rys. 1. Ujęcia zakresu diagnostyki systemów

Dr hab. AGNIESZKA SZEWCZYK ukończyła w 1975 r. na Politechnice Szczecińskiej studia ekonomiczne, ze specjalnością Organizacja Przetwarzania Danych. Od ukończenia studiów pracuje jako nauczyciel akademicki – obecnie w Instytucie Cybernetyki Ekonomicznej i Informatyki Wydziału Ekonomicznego Uniwersytetu Szczecińskiego. W 1983 r. obroniła pracę doktorską pt. „Metoda wyboru organizacji zbiorów danych w systemach informatycznych zarządzania", a w 1992 roku uzyskała stopień doktora habilitowanego w zakresie ekonomii – organizacji przetwarzania danych – informatyki. Praca habilitacyjna pt. „Diagnostyka systemów informacyjnych – problemy metodologiczne" była plonem wieloletnich studiów nad problematyką systemów informacyjnych i stanowi podstawę dalszej pracy naukowej Autorki.

Celem diagnostyki jest ustalenie faktycznego stanu systemu, a w konsekwencji poprawa tego stanu, zapewniająca jego prawidłowy rozwój. Stwierdzenie to dopuszcza zatem prowadzenie działań diagnostycznych, zarówno w przypadku pojawiających się nieprawidłowości w systemie, jak i przy braku takich sygnałów – niejako w celach profilaktycznych.

Proces formułowania diagnozy obejmuje następujące działania:

- badanie systemu pod kątem przyjętych kryteriów lub wartości wzorcowych; Efektem tego badania może być stwierdzenie prawidłowego stanu systemu bądź wykrycie, czy też potwierdzenie, ewentualnych wadliwych procesów lub błędów w strukturze systemu;
- sformułowanie hipotezy roboczej (tzw. wstępnej diagnozy);
- określenie przyczyn nieprawidłowości w systemie i stopnia ich rozwoju;
- wyszczególnienie funkcji nie wykonywanych przez system wskutek zaistniałych nieprawidłowości,
- weryfikacja hipotezy roboczej – sformułowanie ostatecznej diagnozy, czyli określenie stanu systemu.

Natomiast odnowa systemu to określenie sposobów osiągania pożądanego stanu systemu, z uwzględnieniem warunków rozwoju systemu.

Wymienione działania, mimo że mogą się nawzajem uzupełniać i przenikać, stanowią jednak logiczny ciąg czynności pozostających między sobą w związku przyczynowym.

Rozważając aspekty celowości diagnozowania systemu informacyjnego warto zwrócić uwagę na następujące przesłanki:

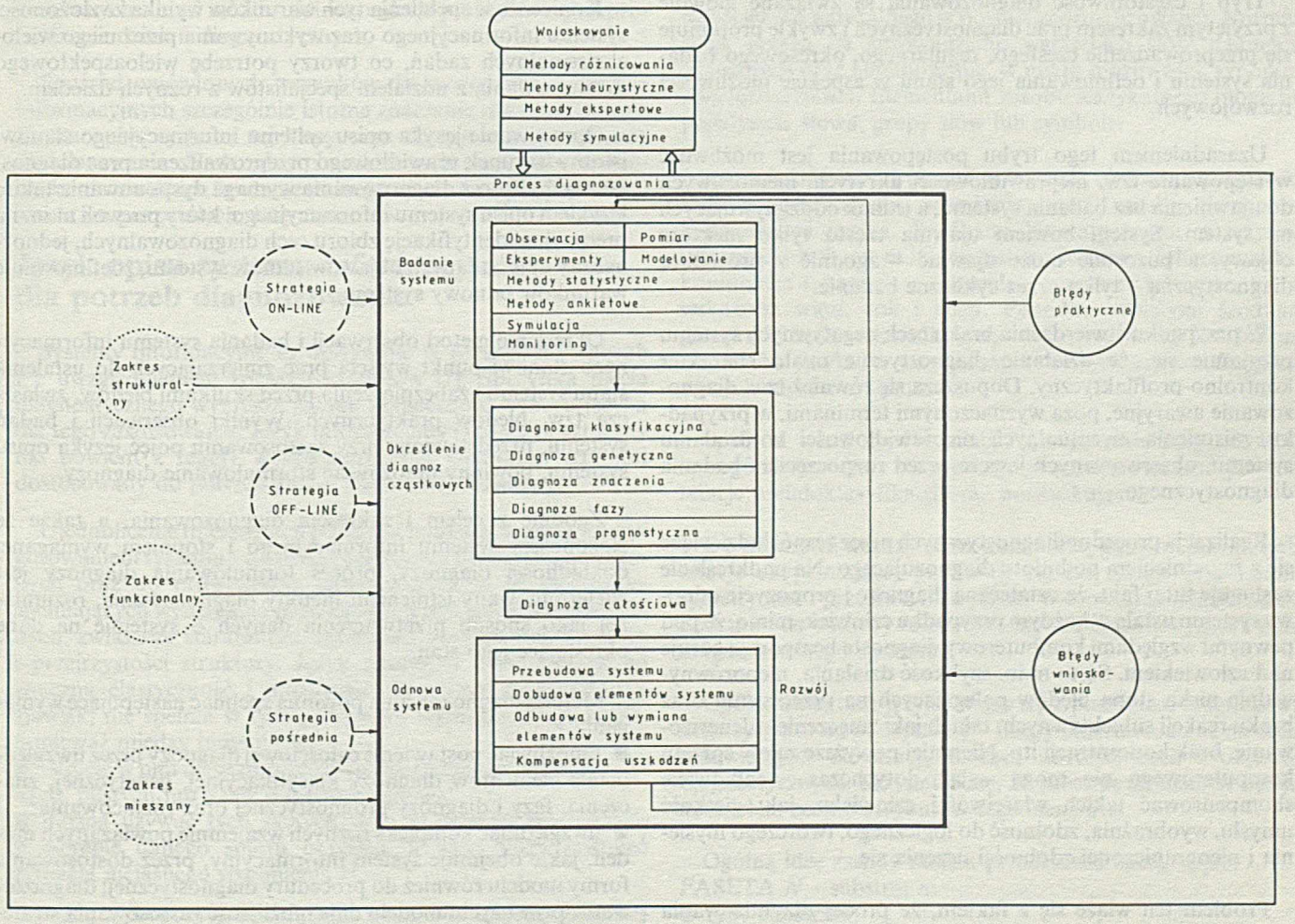
- konieczność zbadania (faktycznego) stanu systemu w aspekcie celu jego działania i struktury,
- zapobieganie wzrostowi negatywnych wartości cech systemu, czyli negatywnego stanu systemu,
- dbałość o niepowodowanie wtórnych błędów w systemie,
- konieczność stwierdzenia, czy system wykorzystuje swoje wszystkie potencjalne możliwości;
- istnienie możliwości zastosowania sposobów odnowy systemu, zapewniających właściwy jego rozwój.

Przeprowadzeniem procedury diagnostycznej powinien być zainteresowany nie tylko użytkownik systemu informacyjnego, który jako klient w warunkach gospodarki rynkowej płaci za usługę wykonaną przez system informacyjny i chce ją uzyskać w jak najlepszej jakości. Zainteresowany jest również właściciel systemu informacyjnego, który w przypadku istnienia konkurencyjnych systemów informacyjnych powinien dbać o szeroko rozumianą jakość systemu, czyli o jego właściwy stan.

Determinanty diagnozowania systemu informacyjnego

System informacyjny, traktowany jako przedmiot badań diagnostycznych, musi być rozpoznawany w określonym stanie, w określonych sytuacjach i z uwzględnieniem przebiegających w nim procesów, w tym także procesów rozwojowych. Jest to możliwe przy wykorzystaniu określonych cech i symptomów, czyli tzw. zestawu charakterystyk systemu.

W tym aspekcie system informacyjny można diagnozować według zakresu strukturalnego, funkcjonalnego lub mieszanego, stosując wybraną strategię diagnozowania (on-line, off-line



Rys. 2. Makroalgorytm procesu diagnostycznego

lub pośrednią). Uwzględnić należy wpływ stopy błędów na proces diagnozowania oraz błędów wnioskowania przy analizie danych diagnostycznych. Powyższe problemy ilustruje rysunek 2.

Przesłanki dające podstawę do rozpoczęcia prac diagnostycznych można przedstawić w formie szczegółowych warunków jakie musi spełnić system informacyjny, aby mógł być diagnozowany. Są to:
- możliwość wyróżnienia zbioru cech diagnostycznych oraz symptomów, a także ich obserwacji i badania systemu pod kątem przyjętych cech,
- istnienie zdefiniowanej struktury systemu, czyli zbioru elementów i określonych w tych elementach relacji,
- istnienie wyróżnionych procesów realizowanych przez system,
- możliwość określenie warunków odnowy systemu (technicznych, organizacyjnych, finansowych i in.).

Warunki diagnozowalności systemu informacyjnego oraz złożoność systemów tego typu wymagają ponadto:
- określenia trybu i częstotliwości diagnozowania,
- realizacji procedur diagnostycznych przez zespół diagnozujący,
- opracowania języka opisu systemu informacyjnego dla potrzeb diagnozowania,
- określenia metod obserwacji i badania systemu informacyjnego,
- opracowania odpowiedniej metody diagnozowania.

Warunki te wymagają bliższego omówienia.

Tryb i częstotliwość diagnozowania są związane głównie z przyjętym zakresem prac diagnostycznych i zwykle proponuje się przeprowadzenie częstego, regularnego, okresowego badania systemu i definiowania jego stanu w aspekcie możliwości rozwojowych.

Uzasadnieniem tego trybu postępowania jest możliwość występowania tzw. nieprawidłowości ukrytych, niemożliwych do ujawnienia bez badania systemu, a jednak oddziaływujących na system. System bowiem ujawnia często tylko niektóre objawy, a pozostałe może ujawnić – zgodnie z procedurą diagnostyczną – tylko przez cykliczne badanie.

W przypadku stwierdzenia braku cech negatywnych systemu przyjmuje się, że działanie diagnostyczne miało charakter kontrolno-profilaktyczny. Dopuszcza się również tzw. diagnozowanie awaryjne, poza wyznaczonymi terminami, w przypadku zaistnienia ewentualnych nieprawidłowości w działaniu systemu, obserwowanych jeszcze przed rozpoczęciem badania diagnostycznego.

Realizacja procedur diagnostycznych przez zespół ludzi wiąże się z zagadnieniem podmiotu diagnozującego. Na podkreślenie zasługuje tutaj fakt, że ostateczną diagnozę i propozycje odnowy systemu ustala w każdym przypadku człowiek, mimo, że pod pewnymi względami komputerowy diagnosta bezspornie góruje nad człowiekiem. Są to m.in. szybkość działania, nieporównywalnie niska stopa błędów polegających na przeoczeniu oraz braku reakcji subiektywnych, takich jak: zmęczenie, zdenerwowanie, brak koncentracji itp. Niemniej powyższe zalety sprzętu komputerowego nie mogą – jak dotychczas – całkowicie skompensować takich właściwości człowieka, jak: giętkość umysłu, wyobraźnia, zdolność do logicznego, twórczego myślenia i nieograniczonej zdolności uczenia się.

Problem ten wiąże się z faktem, że proces diagnozowania należy w swojej ostatecznej fazie do procesów nieustrukturalizowanych. Nie zmienia to sytuacji, że w poszczególnych etapach diagnozowania może korzystać z różnorodnych algorytmów. Ostateczną diagnozę stawia jednak człowiek, który interpretuje uzyskane wyniki i za nie odpowiada. W proces stawiania diagnozy trzeba bowiem wkalkulować nowe, nieznane wcześniej sytuacje decyzyjne. Duże znaczenie w tego typu procesach odgrywają tzw. czynniki subiektywne sytuacji decyzyjnych, takie jak wiedza, doświadczenie, intuicja i osobowość decydenta.

Zgodnie z wynikami badań psychologicznych i socjologicznych można sądzić, że w prowadzonym przez zespół procesie diagnozowania prawdopodobieństwo postawienia dobrej diagnozy będzie stosunkowo duże. Zespół diagnozujący system informacyjny powinien spełniać następujące warunki [2]:
- skład zespołu powinny tworzyć osoby o dużej wiedzy praktycznej i teoretycznej w zakresie przedmiotu badań, a także metod i środków diagnostyki, zwłaszcza komputerowej;
- zespół musi być świadomy niebezpieczeństwa błędów, jakie są możliwe do popełnienia w czasie działań diagnostycznych (nie wykorzystując oczywiście tej świadomości jako powodu do usprawiedliwiania się ze swoich niepowodzeń);
- poszczególni członkowie zespołu powinni pracować w tym samym przedziale czasu, aby obserwacja dotyczyła systemu w tych samych stanach; Powinni także posługiwać się jednakowym językiem opisu systemu, aby bez nieporozumień i nieścisłości określić cechy i objawy;
- działania zespołu powinny być oparte na zasadach sprawnego działania zespołowego, obiektywiźmie, a przede wszystkim na zasadzie podziału funkcji, adekwatnie do rozmiaru i różnorodności prac.

Konieczność spełnienia tych warunków wynika ze złożoności systemu informacyjnego oraz wykonywania przez niego wieloekstremalnych zadań, co tworzy potrzebę wieloaspektowego diagnozowania z udziałem specjalistów z różnych dziedzin.

Opracowanie języka opisu systemu informacyjnego stanowi istotny warunek prawidłowego przeprowadzenia prac diagnostycznych. Proces diagnozowania wymaga dysponowania takim językiem opisu systemu informacyjnego, który pozwoli m.in. na precyzyjną identyfikację zbioru cech diagnozowalnych, jednoznaczny opis badanych objawów stanów systemu, zdefiniowanie warunków odnowy systemu.

Określenie metod obserwacji i badania systemu informacyjnego stanowi punkt wyjścia prac zmierzających do ustalenia stanu systemu i zabezpieczenia przed skutkami błędów, zwłaszcza tzw. błędów praktycznych. Wyniki obserwacji i badań systemu, przedstawione przy zastosowaniu pojęć języka opisu systemu, powinny umożliwiać sformułowanie diagnozy.

Zgodnie z celem i zakresem diagnozowania, a także ze złożonością systemu informacyjnego i stopniem wymaganej dokładności diagnozy, proces formułowania diagnozy jest zdeterminowany istnieniem metody diagnozowania, rozumianej jako sposób przetworzenia danych o systemie na dane określające jego stan.

Metoda diagnozowania powinna spełniać następujące wymagania:
- umożliwiać postawienie całościowej diagnozy przez uwzględnienie elementów diagnozy klasyfikacyjnej, genetycznej, znaczenia, fazy i diagnozy prognostycznej oraz różnicowania;
- uwzględniać kompleks różnych wzajemnie powiązanych modeli, jakie obejmuje system informacyjny, przez dostosowanie formy modelu również do procedury diagnostycznej; diagnozowanie przy użyciu modelu daje możliwość zastosowania strategii diagnozowania typu on-line, tj. strategii umożliwiającej

prowadzenie działań diagnostycznych podczas funkcjonownia systemu;
- umożliwiać diagnozowanie systemu informacyjnego w aspekcie jego struktury i realizowanych procesów, gdyż dopiero takie ujęcie może dać stosunkowo pełny obraz systemu przez diagnostykę strukturalno-funkcjonalną, otwierającą możliwości rozwoju systemu;
- mieć pożądane cechy procedur diagnostycznych tj. celowość, zgodność wewnętrzną, przejrzystość struktury, wykonalność, prostotę i podatność na zmiany,
- zapewniać diagnozowanie systemu w możliwie najkrótszym czasie; spełnienie tego wymagania wydatnie wspiera zastosowanie technologii komputerowej [3].

Szczegółowy zakres powyższych prac związanych z ustalaniami tzw. „środowiska diagnostycznego", powinien być ustalony każdorazowo przed przystąpieniem do diagnozowania danego typu systemu. Wskazane jest uwzględnienie przy tym takich uwarunkowań, jak:
- cel diagnozowania, związany z przyjętym zakresem diagnostyki,
- złożoność systemu,
- stopień wymaganej dokładności diagnozy,
- dopuszczalny fundusz czasu na diagnostykę (zwłaszcza przy diagnostyce typu off-line),
- dysponowany czas na odnowę systemu,
- liczba oraz jakość sygnalizowanych nieprawidłowości,
- dopuszczalny koszt działań diagnostycznych,
- możliwości oraz kwalifikacje personelu,
- warunki otoczenia.

Właściwe określenie tych czynników oraz wykorzystanie związanych z nimi możliwości warunkuje prawidłowe przeprowadzenie działań diagnostycznych.

Spośród omówionych warunków diagnozowania systemów informacyjnych szczególnie istotne znaczenie ma opracowanie języka opisu systemu tej klasy oraz skonstruowanie metody diagnozowania. Nadrzędny charakter tych elementów wymusza konieczność ich fundamentalnego ujęcia oraz podporządkowania im pozostałych determinantów.

Język opisu systemu informacyjnego dla potrzeb diagnozowania

Systemy informacyjne są opisywane w różnorodny sposób i z uwzględnieniem różnych kryteriów [1]. Specyfika ujęcia diagnostycznego wymusza jednak konieczność wyboru języka dla tego zastosowania. Język informacyjny jako język sztuczny ma bowiem z reguły wyspecjalizowany charakter, tzn. jest dostosowany do potrzeb jednej konkretnej dziedziny.

Ujednolicenie interpretacji pojęć jest punktem wyjścia – przygotowaniem systemu – do przeprowadzenia badań diagnostycznych oraz do prowadzenia dalszych prac w tej dziedzinie, spełniających wymagania prawidłowych procesów diagnostycznych, zwłaszcza staranności, zgodności wewnętrznej, prostoty i przejrzystości struktury. Język naturalny, choć cechuje go znaczna elastyczność i możliwość wyrażania różnorodności zjawisk, nie spełnia w tym przypadku wymagań diagnostyki, ponieważ między innymi [5]:
- istnieje w nim niebezpieczeństwo tworzenia tzw. nazw pozornych, pustych lub nazw o intencji ogólnej,
- można używać tych samych słów w różnych znaczeniach w obrębie danego splotu określeń, a więc może zaistnieć sytuacja niejasnego rozumienia słów.

Jedną z propozycji rozwiązania problemu niejednoznacznej lub rozmytej ostrości pojęć (narzędzi) opisu systemu jest koncepcja tzw. języka informacyjnego. Jest on przeznaczony do optymalizycji podstawowych funkcji języka, tj. funkcji komunikacyjnej i służy do jednoznacznego zapisania wszelkiego rodzaju informacji w celu ich późniejszego wykorzystania.

Analiza cech, jakie powinien mieć język przeznaczony do opisu systemu informacyjnego, prowadzi do wniosku, że adekwatny do potrzeb diagnozowania tej klasy systemów będzie język deskryptorowy typu fasetowego, oparty na indeksowaniu dokonywanym na podstawie słownika.

Istotne są takie właściwości tego typu języka, jak:
- duża przydatność praktyczna, związana z oparciem tego języka na terminach języka naturalnego,
- łatwe wprowadzanie nowych terminów, co czyni ten język otwartym na opis nowych cech i kryteriów,
- możliwość dołączenia słownika, gwarantująca kontrolowane narastanie słownictwa specjalistycznego,
- możliwość występowania kilku klas naczelnych oraz wielu ciągów hierarchicznych, co jest korzystne przy prowadzeniu działań diagnostycznych systemów informacyjnych różnych klas i z uwzględnieniem różnych kryteriów,
- możliwość doboru faset z punktu widzenia użyteczności przy diagnozowaniu; faseta jest tu rozumiana jako grupa terminów odpowiadająca wspólnej charakterystyce; układ faset powoduje, że opis systemu jest ukierunkowany tematycznie.

Ustalenie typu języka wymaga określenia jego zakresu semantycznego. W tym przypadku zakres ten jest wyznaczany przez zbiór cech diagnostycznych.

Podstawowy zasób leksykalny powinien zawierać niezbędne słowa języka naturalnego w postaci znormalizowanej. Normalizacja polega na sprowadzeniu wszystkich słów najczęściej używanych oraz eliminowanie, w miarę możliwości, synonimii i wieloznaczności. Elementami zasobu leksykalnego mogą być pojedyncze słowa, grupy słów lub symbole.

Język informacyjny powinien też mieć określoną gramatykę, która stanowi środek wyrażania relacji między elementami leksykalnymi języka. Środkami gramatycznymi w językach informacyjnych są m.in. operatory logiczne (alternatywa v, koniunkcja ^, negacja ~, łącznik -, operatory relacji $<$, $>$, $=$, wskaźniki więzi, roli i wagi. Ponadto swoistymi środkami gramatyki będą wskaźniki pasywacji, aktywizacji i stabilizacji elementów języka, a szczególnie cech i symptomów diagnozowanych stanów systemu.

Spośród wielu typów relacji, dla potrzeb diagnozowania szczególne znaczenie mają relacje skojarzeniowe (asocjacyjne), relacje pseudoklasyfikacji oraz porządkujące.

Do powiązań między elementami leksykalnymi języka (jako relacje uzupełniające) przyjmuje się też relacje przechodnie, symetryczne i przeciwsymetryczne. Wymienione relacje tworzą podstawę do budowy faset – głównych elementów języka opisu diagnozowanego systemu.

Ogólna struktura języka wiąże się z podziałem diagnozowanego systemu na podsystemy – fasety ukierunkowane na cechy i zawierające substraty, niosące wartości cech i objawów diagnostycznych. Przyjmuje się, że substrat to element będący nosicielem cech – atrybutów tego elementu.

Ogólną ideę zasady można przedstawić następująco:
FASETA N – substrat n:
- typy substratu,
- powiązania wewnętrzne substratów,

- powiązanie zewnętrzne substratów,
- powiązania z fasetą kwalifikacji diagnostycznych,
- powiązania z procesami informacyjnymi.

Należy podkreślić, że układ faset musi być koherentny zarówno na poziomie najwyższym, tj. faset, jak i na poziomach substratów i powiązań. Łatwo zauważyć, że już sama faseta spełnia funkcję czynnika porządkującego, umożliwiając w ten sposób celowe grupowanie cech.

Substraty w fasetach diagnostycznych, w połączeniu z oceną dokonywaną za pomocą tzw. fasety kwalifikacji diagnostycznych, są traktowane jako elementy o potencjalnej wartości diagnostycznej. Inaczej są to elementy systemu informacyjnego, na które należy zwrócić szczególną uwagę jako na możliwie niedomagające miejsca w systemie. Ich niewłaściwa budowa, funkcjonowanie, wybór niewłaściwego typu itp. mogą być potencjalnym źródłem nieprawidłowości w systemie bądź stanowić objaw innych nieprawidłowości.

W prezentowanym języku opisu systemu informacyjnego ustalono następujące fasety substratów diagnostycznych:
ZASOBY INFORMACYJNE – Z,
PROCEDURY TECHNOLOGICZNE – T,
CZŁOWIEK – C,
SPRZĘT – S,
OSNOWA SYSTEMU – O,
i jako uzupełnienie
KWALIFIKACJE DIAGNOSTYCZNE – D.

Opis systemu informacyjnego w układzie fasetowym powinien zawierać:
- numer substratu,
- nazwę substratu,
- typ substratu,
- powiązania wewnętrzne (w ramach fasety),
- powiązania zewnętrzne (z innymi fasetami),
- powiązania z fasetą kwalifikacji diagnostycznych,
- powiązania z procesami informacyjnymi.

Natomiast struktura fasety *KWALIFIKACJE DIAGNOSTYCZNE* została sprowadzona do:
- numeru substratu,
- nazwy substratu,
- łańcuchów powiązań wewnętrznych.

Substraty diagnostyczne w ramach fasety zawierają cechy sytuacyjne, działalności, struktury i osiągnięć. Każdy substrat otrzymuje też propozycję określenia typu z punktu widzenia potrzeb diagnostyki. Typologia ta powinna pomóc zespołowi przeprowadzającemu działania diagnostyczne w rozróżnianiu poszczególnych atrybutów i w nadawaniu im roli diagnostycznej.

Przyjmuje się następujące typy substratów:
- ewolujący – *A*;
- dynamiczny – *B*;
- podstawowy – *C*;
- strukturalny – *D*;
- naturalny – *E*;
- przypadkowy – *NA*,
- statyczny – *NB*,
- pochodny (wtórny) –*NC*,
- funkcjonalny – *ND*,
- wywołany *NE*.

Proponując takie rozróżnienie typów substratów diagnostycznych przyjmuje się jednocześnie, że wszystkie one mają charakter aktualno-przyczynowy, gdyż będą tworzone z nich tzw. łańcuchy przyczynowo-skutkowe. Nie jest bowiem możliwe ustalanie przyczyn i skutków obserwowalnych jako objawy, gdyż niektóre z nich są z kolei przyczynami następnych objawów. Sztywny podział zubożyłby opis systemu.

Wewnątrz każdej fasety znajdują się substraty jakościowe, ilościowe i kosztowne. Wśród tych trzech grup mieszczą się bowiem wszystkie potencjalne cechy systemu informacyjnego, które należy uwzględnić realizując proces diagnozowania. Przewiduje się, że takie ujęcie wewnętrznej struktury fasety będzie pomocne dla zespołu diagnozującego, zwłaszcza przy podejmowaniu decyzji dotyczących przyjętych kryteriów diagnozowania.

Format faset diagnostycznych zawiera oprócz nazw substratów i ich typów także powiązania między substratami wewnątrz fasety oraz z substratami innych faset. Pozwala to na zapis związków między substratami z zastosowaniem elementów gramatyki języka fasetowego i na uwzględnienie zjawiska kontyngencji, czyli współzależności między substratami, z których przynajmniej jeden jest jakościowy.

Uzupełnienie powiązań wewnętrznych i zewnętrznych stanowią powiązania substratów z fasetą kwalifikacji diagnostycznych, wskazujące sposób wykorzystania jej zawartości do określania typów, nasilenia, czy też jakości objawów diagnostycznych.

Fasety zawierają również powiązania substratów diagnostycznych z realizowanymi w systemie procesami informacyjnymi. Powiązania te wskazują, że określony substrat (w zależności od jego wartości), ma wpływ na prawidłowy lub nieprawidłowy przebieg tego procesu. Przyjmuje się następujące typy procesów informacyjnych:
- tworzenie danych – *D*,
- gromadzenie danych – *G*,
- przetwarzanie danych – *P*,
- magazynowanie danych – *M*,
- emitowanie danych – *E*.

Poniżej przedstawiono opis zawartości tematycznej wyodrębnionych faset oraz przykładowe tabele ilustrujące sposób tego opisu w notacji opracowanego języka informacyjnego.

Faseta *ZASOBY INFORMACYJNE – Z* obejmuje wyodrębnione substraty dotyczące danych, które są wprowadzane do systemu informacyjnego, ulegają w nim przetworzeniu oraz są z niego wyprowadzane. Dane te są grupowane w zbiory danych wejściowych, wyjściowych i operacyjnych.

Umieszczenie wszystkich wyróżnionych zbiorów w jednej fasecie umożliwia dignostykę całego masywu danych w systemie informacyjnym oraz zapobiega nadmiernej redundancji substratów. Faseta ta ma w penym sensie charakter uniwersalny, ponieważ znajdują się w niej zarówno substraty możliwe do wykorzystania w systemach informatycznych, jak i substraty odpowiednie dla systemów informacyjnych funkcjonujących bez sprzętu komputerowego.

Faseta *PROCEDURY TECHNOLOGICZNE – T* obejmuje zagadnienia funkcjonowania systemu informacyjnego w aspekcie procedur technologicznych – ich cech, czasu trwania i elementów umożliwiających ich wykonanie. Niektóre substraty dotyczą wyłącznie systemów obsługiwanych przez sprzęt komputerowy (np. typ kompilatora, typ systemu operacyjnego), lecz większość z nich ma zastosowanie również do diagnozowania systemu informacyjnego, opartego na tradycyjnych procesach technologicznych (tabela 1).

Faseta *CZŁOWIEK – C*. Wyodrębnienie substratów tej fasety ma znaczenie szczególne, gdyż zwraca uwagę na wpływ człowieka na stan systemu informacyjnego. Jest to obszar najtrudniej diagnozowalny, poddający się prawie wyłącznie charakterystyce jakościowej, niewymiernej.

Tabela 1. Faseta – PROCEDURY TECHNOLOGICZNE – T

Numer substratu	Nazwa substratu	Typ substratu	Powiązania wewnętrzne	Powiązania zewnętrzne	Powiązania z fasetą D	Powiązania z procesami informacyjnymi
1	Typ technologii przetwarzania danych	A-NB-C-D-E	16-12-3-2	Z1-S1	14-20-24-42-45	P-G-E
2	Tryb dostępu do danych	A-NB-C-ND-E	3-7-16	Z2-S7-S10-S11	37-32	P-E
3	Metoda wyszukiwania danych	A-NB-C-ND-E	2-7-9-11	Z1-Z12-T18-S7-S9-S18	36-42-32	P-E
4	Typ języka programowania	A-NB-C-D-E	8-7-5-6-10	S1-S20-C1	15-22-24-27-28-33-40	P-E
5	Typ techniki programowania	A-NB-NC-D-E	4-9	C1-C3-010	33-15-37-42-40	P-E
6	Metoda tworzenia oprogramowania	A-NB-NC-C-E	7-4-10	C1-C3-02-010	22-39	P-G
7	Typ systemu operacyjnego	A-NB-C-D-E	1-13-14-16	Z14-Z1-Z7-Z9-Z8-S1-S10-09	21-24-27-29-36-37-26	P-M-E
8	Typ kompilatora	A-NB-NC-D-E	4-14-15	S1-S10	21-24-27-36-37 26	P
9	Rodzaj oprogramowania	A-B-NC-ND-E	7-6	Z22-S10-S9-01	15-27	P
10	Zabezpieczenie programowe	NA-NB-NC-D-E	16-7-2	Z11-S19-C1-C2	36-40-58	P
11	Sposób przesyłania danych	A-B-NC-ND-E	7-16-12	Z3-Z5-Z19-Z13-Z20-S2-S3-S4-S6-S17-S18-013-014	14-22-37-43	P-M-E-G
12	Czas realizacji procedury	NA-B-NC-ND-E	7-14-15-8	Z6-Z15-Z21-Z16-S13-S12-S11-S10-C1-C3-C6-C7-C4	17-50-52-58	P-E
13	Cykl przetwarzania	NA-B-NC-ND-E	14-16-7	Z17-S12-S13-S15-S16-S17-S18-C3-C8	49-50-52	P-E-G
14	Natężenie procedur	NA-B-NC-ND-E	7-16-4-9	S21-S10-S9-S11-S13-C1-C3-C8-C4-C7-015-016	50-52-58	D-G-P-M-E

Substraty zawarte w fasecie *C* dotyczą potencjonalnych punktów komunikacji człowieka z systemem informacyjnym. Obrazuje to typowe powiązanie zewnętrzne z pozostałymi fasetami substratów diagnostycznych. Określają one niezbędny zakres oddziaływania i dyspozycyjność człowieka w realcji do systemu.

Grupa substratów fasety *SPRZĘT – S* zawiera charakterystykę sprzętu wykorzystywanego dla potrzeb systemu informacyjnego. Ujęto w niej substraty dotyczące zarówno sprzętu komputerowego, jak i innych rodzajów sprzętu z grupy szeroko rozumianej mechanizacji prac obliczeniowych. Poziom szczegółowości substratów umożliwia charakterystykę sprzętu pod kątem parametrów tzw. mocy obliczeniowej (pojemność pamięci, czas, cykl) jak i jakości sprzętu, liczby niezbędnych elementów, ich typu itp.

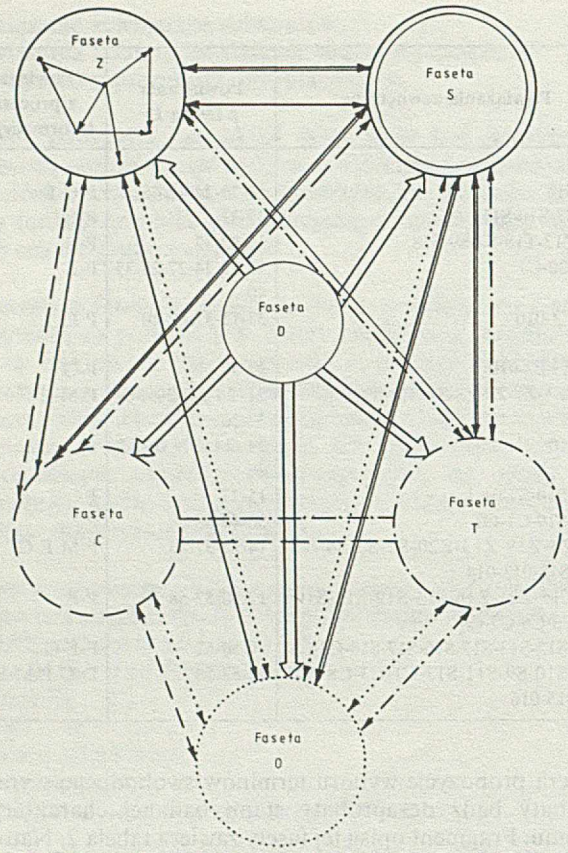
Układ fasety umożliwia całkowite pominięcie jej zawartości w procesie diagnozowania, jeżeli system informacyjny nie jest wspomagany żadnym typem sprzętu lub gdy zespół diagnozujący uzna, że wykorzystywany sprzęt ma odpowiednie parametry jakościowe i ilościowe.

Faseta *OSNOWA SYSTEMU –O* zawiera substraty tworzące strukturę funkcjonalną systemu informacyjnego. Stanowią one w pewnym sensie systezę pozostałych faset i przenikając je tworzą ich swoistą osnowę, wyznaczając podział funkcjonalny elementów systemu, powiązania poziome i pionowe oraz relacje między celem systemu i jego działaniem, a także zgodnością systemu z projektem i metodami wdrażania. Niezbędne uzupełnienie struktury funkcjonalnej fasety *O* tworzą substraty kosztowe i efektywnościowe systemu informacyjnego.

Faseta *KWALIFIKACJE DIAGNOSTYCZNE – D* zawiera substraty pozwalające na stwierdzenie występowania diagnozowanych objawów oraz ocenę stopnia ich natężenia. Zbiór charakterystyk systemu informacyjnego powstaje w wyniku przypisania substratom diagnostycznym konkretnych wartości. Faseta zawiera dwie grupy substratów. Pierwszą grupę tworzą substraty opisujące cechy jakości systemu według ściśle określonych i zdefiniowanych w słowniku terminów. Grupa druga zawiera propozycje wyboru terminów swobodnego wyrażania aprobaty bądź dezaprobaty stanu badanej charakterystyki systemu. Fragment opisu tej fasety zawiera tabela 2. Natomiast rysunek 3 przedstawia diagram współzależności i powiązań między wszystkimi fasetami systemu informacyjnego.

Tabela 2. Faseta – KWALIFIKACJE DIAGNOSTYCZNE – D

Numer substratu	Nazwa substratu	Łańcuchy powiązań wewnętrznych
1	Aktualność	Λ 35 Λ 13 V 38
2	Celowość	V 18 Λ 12 Λ 4 Λ 31 Λ 8 Λ 25 Λ 49
3	Czytelność	Λ 13 Λ 23 Λ 30 V 23
4	Dokładność	Λ 2 V 45 V 34 Λ 58
5	Dyferencyjność	V 15 V 24 Λ 12
6	Dyspozycyjność	Λ 39 Λ 9
7	Efektywność	Λ 42 Λ 46 Λ 47 Λ 48 Λ 51
8	Ekwifinalność	Λ 2 V 31 V 22 Λ 59
9	Elastyczność	Λ 6 V 40
10	Ergonomiczność	Λ 43 Λ 42 Λ 18 Λ 53 Λ 54
11	Integralność	V 41 Λ 19
12	Koherentność	Λ 2 Λ 8 Λ 5
13	Komunikatywność	Λ 3 V 43 Λ 30 Λ 23 Λ 34 Λ 39 Λ 1
14	Konserwowalność	V 38 Λ 39 V 26
15	Modularność	V 5 V 33
16	Niesprzeczność	V 25 Λ 23
17	Niezależność	V 18 Λ 29
18	Niezawodność	V 2 Λ 17 Λ 10 V 37
19	Obiektywność	Λ 4 V 40 Λ 11 Λ 57 V 59
20	Odporność	Λ 32 Λ 37 Λ 50
21	Ogólność	Λ 44 Λ 28
22	Poprawność	Λ 40 V 8 Λ 52
23	Prostota	Λ 3 Λ 13 Λ 16 Λ 34 Λ 55
24	Przenośność	Λ 5 Λ 15 V 42
25	Relewantność	V 16 Λ 2 V 49 Λ 52 Λ 55 Λ 57
26	Reproduktywność	V 28 V 14
27	Rozliczalność	Λ 33 V 36 V 37
28	Rozszerzalność	V 26 Λ 21
29	Samowystarczalność	Λ 37 Λ 17
30	Selektywność	Λ 13 Λ 3 V 58
31	Skuteczność	Λ 2 V 8 V 46
32	Sprawność	Λ 20 Λ 37 Λ 50 Λ 53
33	Strukturalność	V 15 Λ 27



Rys. 3. Diagram współzależności i powiązań między fasetami w systemie informacyjnym

Proponowany język opisu systemu informacyjnego należy do niesformalizowanego typu opisu modelowanej rzeczywistości, ma wysoki poziom deklaracji semantycznej oraz umożliwia werbalizację problemu w języku naturalnym Czyni to z niego praktyczną podstawę opisu systemu, umożliwiającą diagnozowanie jego stanu.

LITERATURA

[1] Flakiewicz W.: Systemy informacyjne przedsiębiorstw i instytucji. PWE, Warszawa 1982
[2] Kozielecki J.: Psychologiczna teoria decyzji. PWN, Warszawa 1977
[3] Morton P.W.: A Model for Planning Effective Information Systems. Journal of Information Image Management 5, 1985
[4] Raport PTI – Cele zastosowań informatyki w zarządzaniu. PTI, Warszawa 1987
[5] Ungurian O.: Elementy teorii języków informacyjnych. PAN – OIN, Warszawa 1976.

W związku z wprowadzeniem powszechnego podatku dochodowego i koniecznością informowania właściwych Urzędów Skarbowych o wszystkich wypłatach zwracamy się do Autorów wysyłających teksty do opublikowania o podawanie następujących danych:

- **nazwisko, imiona (pierwsze i drugie),**
- **imiona: ojca i matki,**
- **miejsce i data urodzenia,**
- **numer identyfikacyjny PESEL (wpisany przez Biuro Meldunkowe do dowodu osobistego – nie mylić z numerem dowodu osobistego!),**
- **dokładny adres (miejsce zameldowania),**
- **adres Urzędu Skarbowego właściwego dla miejsca zamieszkania Autora (bardzo ważne – bez tej informacji nie możemy przekazać honorarium do wypłaty!).**

TERA Spółka Z O. O.

Przedsiębiorstwo Popierania Postępu TERA Spółka z o.o. uprzejmie informuje, że posiadając kilkuletnie doświadczenie w instalacji systemów wspomagających zarządzanie

o f e r u j e :

- opracowanie projektów systemów,
- optymalny kosztowo i rozwojowo dobór sprzętu i oprogramowania,
- instalację systemu u klienta,
- bezpłatnie przez rok konserwację oprogramowania oraz serwis sprzętu wraz z doradztwem techniczno-eksploatacyjnym,
- szybką dostawę uzupełnień konfiguracji lub sprzętu mikrokomputerowego niezależnego od systemu, w tym mikrokomputerów ALR firmy Wearnes Technology.

Wszelkie dodatkowe informacje uzyskają Państwo codziennie oprócz niedziel w Biurze Handlowym

40-025 Katowice, ul. Wojewódzka 31
tel. (faks): 155-26-72, teleks: 315448 tera pl

PAMIĘTAJ! Instalacje XENIX/NOVELL/PC MOS 386 oraz serwis to nasza specjalność – ZAPRASZAMY, ponieważ czterech lat doświadczeń nigdzie nie kupisz.

0/19/90

meditronik SPÓŁKA z o.o.

- CZĘŚCI ELEKTRONICZNE
- KOMPUTERY PS/1,PS/2
- DRUKARKI HP
- INSTALACJE SIECI KOMPUTEROWYCH

Partnerzy handlowi: ANALOG DEVICES, ITT, MOTOROLA, SAMSUNG, TELEFUNKEN i inni

UMC UNITED MICROELECTRONICS CORPORATION

- UKŁADY PAMIĘCI
- UKŁADY KOMPUTEROWE
- UKŁADY KOMUNIKACYJNE I KOMERCYJNE

PRZEDSTAWICIELSTWO

HEWLETT PACKARD COMPONENTS

- TRANSOPTORY
- WSKAŹNIKI ŚWIETLNE
- WYŚWIETLACZE LED
- PRODUKTY KODÓW KRESKOWYCH
- KONTROLERY I CZUJNIKI RUCHU
- TECHNIKA ŚWIATŁOWODOWA
- ELEMENTY W.CZ. I MIKROFALOWE
- PODZESPOŁY DO MONTAŻU POWIERZCHNIOWEGO (SMD)

DYSTRYBUCJA

BOURNS

- POTENCJOMETRY TRIMPOT
- HYBRYDY REZYSTOROWE
- REZYSTORY SUBMINIATUROWE
- BEZPIECZNIKI MULTIFUSE
- POTENCJOMETRY PRECYZYJNE
- POTENCJOMETRY PANELI CZOŁOWYCH I KODERY
- CEWKI I TRANSFORMATORY
- CZUJNIKI CIŚNIENIA, POŁOŻENIA I PRZYŚPIESZENIA

PMI

DYSTRYBUCJA

IBM Business Partner

meditronik sp. z o.o.

00-194 Warszawa,ul. Dzika 4
tel. (02) 6352263, 6352264
fax (02) 6352195, tlx 816075

SO/575/91

RAFAŁ KRAJEWSKI
Instytut Informatyki Teoretycznej
i Stosowanej
PAN
Gliwice

Dlaczego NIS?

Dynamiczny rozwój komputerowych rozwiązań sieciowych, zarówno sprzętowych jak i programowych, oprócz udostępniania użytkownikom coraz nowszych jakościowo usług związanych z przesyłaniem zbiorów i korzystaniem ze wspólnych zasobów sieci komputerowej, niesie ze sobą problemy dotyczące zarządzania tymi sieciami. Pod pojęciem zarządzania rozumiemy czynności wykonywane przez administratora systemu sieciowego (takie jak zarządzanie informacjami na dyskach lub taśmach czy rozbudowywanie istniejących konfiguracji sieciowych), które mają na celu udostępnianie użytkownikom produktywnie działających i włączonych do sieci komputerów. O ile zarządzanie takie w przypadku sieci o prostych architekturach nie jest zbyt skomplikowane, o tyle w przypadku sieci rozbudowanych architektonicznie i systemowo może okazać się trudne i czasochłonne. Konieczne staje się wówczas stosowanie mechanizmów wspomagających.

Jednym z takich mechanizmów, który istotnie poprawia efektywność pracy administratora systemu sieciowego jest NIS – *The Network Information Service*, znany dawniej pod nazwą *Yellow Pages* (Żółte Strony).

Tworzenie, rozbudowywanie oraz poprawne funkcjonowanie lokalnych (choć nie tylko) sieci komputerowych opartych na systemach unixopodobnych wymaga zachowania jednoznaczności pewnych systemowych zbiorów we wszystkich komputerach w tych sieciach. Zachowanie jednoznaczności tych zbiorów gwarantuje poprawne komunikowanie się komputerów między sobą, natomiast NIS jest usługą, która tę jednoznaczność zapewnia. Zbiory zarządzane przez NIS zawierają przede wszystkim informacje o charakterze sieciowym i są przechowywane przez UNIX w katalogu */etc*; są to m.in.: */etc/hosts*, który przechowuje adresy sieciowe i nazwy sieciowe wszystkich głównych komputerów w sieci lub */etc/passwd*, który przechowuje informacje o wszystkich użytkownikach w każdym z tych komputerów (dodajmy, że liczba zbiorów standardowo zarządzanych przez NIS jest znacznie większa).

Bez stosowania mechanizmu NIS, jednoznaczność i spójność tych zbiorów spoczywają na odpowiedzialności administratora sieci. Jest to kłopotliwe i czasochłonne co najmniej z dwóch powodów: po pierwsze – zwiększa się ryzyko popełniania przez niego błędów prowadzących do niespójności tych zbiorów, po drugie – często jest utrudniony fizyczny dostęp do wszystkich komputerów w sieci ze względu na ich wzajemne znaczne oddalenie.

Idea NIS

Koncepcja działania usługi NIS jest oparta na podziale komputerów przyłączonych do sieci na dwie grupy: usługodawców (ang. *servers*) i usługobiorców (ang. *clients*). Podział ten jest dokonywany przez administratora sieci w ramach określonego obszaru w sieci zwanego dziedziną NIS-ową (przykładem takiej dziedziny może być zbiór kilku czy kilkunastu wybranych w sieci komputerów eksploatowanych przez użytkowników mających wspólnie określone cele). W danej dziedzinie usługodawcy udostępniają do odczytu usługobiorcom informacje zawarte w tzw. mapach NIS-owych. Mapy te z kolei odpowiadają zarówno pod względem nazwy, jak i pod względem treści zbiorom zawartym w katalogu */etc*.

Tworzenie i modyfikowanie map NIS-owych ma charakter hierarchiczny. Są one tworzone i uaktualniane tylko w jednym, wyróżnionym przez administratora sieci komputerze zwanym nadrzędnym usługodawcą (ang. *master-server*). Następnie mapy te są automatycznie (bez ingerencji administratora) rozsyłane do wyróżnionych komputerów zwanych podrzędnymi usługodawcami (ang. *slave-servers*). Z tak rozmieszczonych w sieci map NIS-owych (a konkretnie z informacji w nich zawartych) korzystają komputery-usługobiorcy tzw. klienci (ang. *clients*).

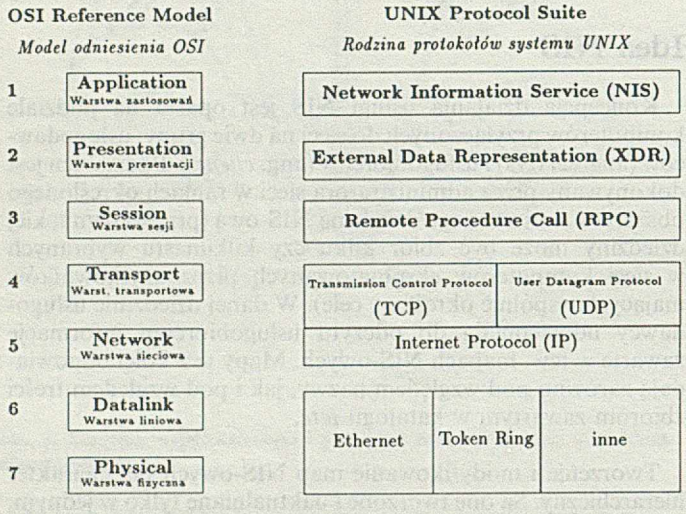
Hierarchiczna struktura propagowania map do usługodawcy nadrzędnego do usługodawców podrzędnych, a następnie udostępnianie tych map usługobiorcom, zapewnia zachowanie zgodności systemowych zbiorów we wszystkich komputerach przyłączonych do sieci.

Mechanizm działania

Struktura działania usługi sieciowej NIS jest oparta na siedmiowarstwowym modelu komunikacyjnym zdefiniowanym przez ISO – *International Organization for Standardization*, znanym jako Model Odniesienia Systemu Otwartego (*Open Systems Interconnection reference model* – OSI). Model OSI opisuje system komputerowy, który komunikuje się z innym systemem komputerowym przez procesy wykonywane we wzajemnie identyfikowanych dyskretnych warstwach. Rysunek ilustruje siedem zdefiniowanych przez ISO warstw modelu odniesienia OSI oraz odpowiadające im protokoły w systemie UNIX, odpowiedzialne za realizację usługi NIS.

Dwie dolne warstwy modelu realizują funkcje komunikacyjne na poziomie najmniejszych jednostek informacyjnych w sieci (bitów i ramek). Są one najczęściej realizowane sprzętowo, zgodnie z obowiązującymi standardami komunikacyjnymi, takimi jak Ethernet czy Token Ring. Następne dwie warstwy wchodzą w skład rodziny protokołów komunikacyjnych TCP/IP będących integralną częścią prawie wszystkich imple-

mentacji systemu operacyjnego UNIX. Ich funkcją jest dostarczanie użytkownikowi tzw. bezbłędnego kanału wirtualnego łączącego dwa komputery przyłączone do sieci. Warstwa sesji w usłudze sieciowej NIS jest realizowana przez protokół *Remote Procedure Call* – RPC. Jest to zbiór procedur zdalnie wywoływanych przez procesy usługodawców lub usługobiorców. Proces koordynujący wykonywanie tych procedur jest znany w UNIX-ie jako tzw. demon (ang. *demon*). Jest to proces, który jest wykonywany w tle i nie jest związany z żadnym konkretnym użytkownikiem. Warstwa prezentacji jest realizowana przez protokół *External Data Representation* – XDR i odpowiada za jednolity sposób przedstawiania przesyłanych informacji, zrozumiały dla wszystkich uczestników sesji. Warstwę zastosowań stanowi sieciowa usługa informacyjna – NIS.



Warstwy modelu OSI oraz odpowiadające im warstwy protokołów realizujących usługę NIS

Uruchomienie usługi NIS w środowisku sieciowym rozpoczyna się od ustanowienia dziedzin NIS-owych i określenia, które komputery w sieci będą należały do określonej dziedziny. Nazwa dziedziny jest ustalana w każdym komputerze w zbiorze */etc/rc.local* na podstawie zawartości zbioru */etc/defaultdomain* podczas ładowania systemu lub podczas pracy systemu za pomocą zlecenia *domainname*. Administrator sieci w każdej dziedzinie przydziela jednemu komputerowi funkcję usługodawcy nadrzędnego (*master-server*). Tylko na tym komputerze tworzone są mapy NIS-owe i tylko na tym komputerze mogą być one zmieniane. Generowanie nowych map następuje po uruchomieniu przez administratora sieci programu *ypinit*, który z kolei używa programów: *make, Makefile* oraz *makedbm*. Mapy NIS-owe po ich utworzeniu są przechowywane w katalogu */etc/yp/<domainname* usługodawcy nadrzędnego. Strukturalnie mapy są zbiorami zawierającymi uporządkowane dane, które usługodawcy udostępniają usługobiorcom. Każda mapa zawiera zbiór kluczy oraz odpowiadające tym kluczom wartości. Niektóre mapy są przechowywane przez usługodawcę podwójnie: jest to uzależnione od tego, jaki atrybut w danej mapie jest kluczem (np. przez usługę NIS są tworzone dwie mapy związane ze zbiorem */etc/hosts: hosts.byaddr* i *hosts.byname*; pierwsza mapa zawiera jako klucze adresy sieciowe – *IP addresses* – wszystkich głównych komputerów przyłączonych do sieci, natomiast jako wartości – nazwy odpowiadające tym adresom; druga mapa – odwrotnie). Mapy po utworzeniu przez NIS (a dokładniej po przekształceniu przez *makedbm*) są przechowywane w specjalnym formacie (*ndbm*) za pomocą dwóch zbiorów – *mapa.dir* i *mapa.pag*; pierwszy z nich zawiera mapę bitową, drugi – wszystkie dane.

Administrator sieci w danej dziedzinie, oprócz wyznaczenia komputera pełniącego funkcję nadrzędnego usługodawcy (*master-server*), może także wyznaczyć usługodawców podrzędnych (*slave-servers*) którzy mają w katalogach */etc/yp/<domainname>* identyczne kopie map, jak usługodawca nadrzędny.

Jeżeli w jakimkolwiek momencie administrator systemu stworzy lub zmieni mapy znajdujące się w komputerze usługodawcy nadrzędnego, to system NIS zapewni propagację tych zmian do wszystkich usługodawców podrzędnych. Proces propagacji map jest pierwotnie inicjowany przez program *ypinit*, natomiast sam przesył realizują programy: *ypxfr* oraz *yppush*; *yppush* jest zawsze uruchamiany na komputerze *master-server* zaraz po tym, jak mapy na tym komputerze uległy modyfikacji. Po stworzeniu przez usługę NIS listy wszystkich usługodawców podrzędnych w danej dziedzinie, program *yppush* wysyła tam usługodawcom żądanie sprowadzenia sobie map NIS-owych. Po potwierdzeniu tego żądania, usługodawcy podrzędni uaktywniają program *ypxfr*, który im sprowadza mapy.

Z tak rozpropagowanych po sieci map korzystają komputery usługobiorcy (w usłudze NIS usługobiorcy nie mają swoich map). Komunikacja między usługodawcą, a usługobiorcą jest oparta na kilku komunikujących się ze sobą demonach: po stronie usługodawcy administrator sieci uaktywnia *ypserv*, natomiast po stronie klienta – *ypbind*. Proces *ypserv* udostępnia klientom informacje przechowywane w mapach. Aby programy klientów mogły z tych informacji korzystać, muszą zrealizować następujące etapy umożliwiające połączenie:

- program uruchamiany po stronie klienta w celu uzyskania informacji przechowywanych w mapach NIS-owych, przepytuje demon *ypbind* o wolnego usługodawcę, który może tej informacji udzielić,
- zadaniem demona *ypbind* jest rozgłoszenie komunikatu po całej sieci z żądaniem dostępu do map należących do usługodawcy i zapamiętanie adresu tego usługodawcy, który odpowie jako pierwszy,
- demon *ypbind* informuje następnie procesy usługobiorców, z którym usługodawca nawiązał połączenie; proces usługobiorcy wysyła żądanie dostępu do map tego usługodawcy, z którym *ypbind* się połączył,
- demon *ypserv*, uruchomiony na komputerze usługodawcy, przechwytuje żądanie wysłania informacji zawartej w podanej mapie, a następnie ją wysyła.

Powiązania między określonymi usługodawcami udzielającymi informacji, a usługobiorcami odczytującymi te informacje nie są ustanawiane na stałe i w zależności od stanu pracy systemu sieciowego mogą zmieniać się w czasie.

Wykorzystanie mechanizmów sieciowej usługi informacyjnej

Ułatwienia wynikające ze stosowania sieciowej usługi informacyjnej NIS zilustrujemy na przykładzie map związanych ze zbiorami */etc/hosts* i */etc/passwd*. Dołączenie komputera do istniejącego środowiska sieciowego w przypadku nie zastosowania usługi NIS wymaga modyfikacji zbioru */etc/hosts* na każdym komputerze znajdującym się w sieci (modyfikacja tego zbioru polega na dodaniu do niego adresu sieciowego oraz nazwy sieciowej przyłączonego komputera). Oczywiste jest, że w przypadku sieci komputerowej o dużej liczbie komputerów, często oddalonych od siebie, operacja taka jest czasochłonna. Zwiększa się także prawdopodobieństwo wystąpienia błędów prowadzących do niespójności tych zbiorów, co z kolei prowadzi do niepoprawnej komunikacji między komputerami. Informacje przechowywane przez zbiór */etc/hosts* są wykorzystywane m.in. przez takie aplikacje sieciowe, jak: FTP – *File Transfer Protocol*, która realizuje przesył plików. *Telent*, która umożliwia sesję zdalnego terminala, NFS – *Network File System*, która

umożliwia dostęp do sieciowego systemu rozproszonych plików, czy program *Ping* pełniący funkcje diagnostyczne w sieci.

Zastosowanie usługi NIS ogranicza konieczność modyfikacji do jednego zbioru tylko na jednym, wyróżnionym komputerze nadrzędnym. Po poddaniu edycji tekstowej zbioru */etc/hosts* na tym komputerze, administrator systemu uaktywnia usługę NIS, która automatycznie aktualizuje odpowiadającą zbiorowi */etc/hosts* mapę (lub tworzy ją, jeżeli usługa NIS nie była wcześniej uruchomiona), a następnie propaguje jej zaktualizowaną kopię do pozostałych podrzędnych usługodawców w sieci. Tak rozpropagowana mapa jest udostępniania przez usługodawców nadrzędnych i podrzędnych do odczytu wszystkim komputerom znajdującym się w sieci. Od tej pory programy uruchamiane na wszystkich komputerach (czyli takie, jak: FTP, Telnet, NFS, Ping), które poprzednio – przed uaktywnieniem usługi NIS – zwracały się do swoich lokalnych zbiorów */etc/hosts*, teraz zwracają się do odpowiadającej zbiorowi */etc/hosts* NIS-owej mapy znajdującej się w komputerach-usługodawcach. Przy czym nie jest ważne dla usługobiorcy, który z usługodawców w danym momencie udziela informacji zawartych w mapach, ponieważ w całej dziedzinie są one identyczne.

Należy zaznaczyć, że administrator sieci może (ale nie musi) wyznaczyć w danej dziedzinie usługodawców podrzędnych (niezbędne minimum dla poprawnego działania usługi NIS, to jeden usługodawca nadrzędny i klienci). Obecność jednak wielu usługodawców w dziedzinie zmniejsza czas oczekiwania na uzyskanie dostępu do map, przez wielu klientów w tym samym momencie, co znacznie poprawia jakość usług świadczonych przez NIS.

Administrator sieci może także określać częstotliwość, z jaką poszczególne mapy usługodawców podrzędnych są aktualizowane. I to niezależnie od tego, czy usługodawca nadrzędny je zmodyfikował, czy nie. Jest to szczególnie przydatne w przypadku, gdy jeden z usługodawców podrzędnych jest wyłączony (użytkownik nie pracuje na tym komputerze), a następnie załącza ten komputer. Może się wówczas okazać, że usługodawca ten nie zawiera aktualnych kopii map, co mogłoby prowadzić do błędnego informowania klientów.

Omówmy teraz przypadek dodawania nowego użytkownika do systemów operacyjnych UNIX, pracujących w komputerach przyłączonych do sieci. Gdy nie zastosowano usługi NIS i gdy dany użytkownik chciałby być rozpoznawany we wszystkich komputerach sieci, administrator tej sieci na każdym komputerze musiałby poddać edycji tekstowej zbiór */etc/passwd*, dopisując do niego wiersz z informacji o dodawanym użytkowniku. Przy zastosowaniu usługi NIS modyfikacja tego zbioru jest przeprowadzona tylko na jednym komputerze *master-server*. Co więcej, zakładanie nowego hasła, obowiązującego we wszystkich komputerach w określonej dziedzinie dla danego użytkownika jest też dokonywane na komputerze *master-server*(z poziomu *root'a*). Użytkownik powinien tylko wiedzieć, że hasło zmienia się zleceniem *yppasswd*, a nie, jak dotychczas zleceniem *passwd*. Po zmodyfikowaniu zbioru */etc/passwd* na tym samym komputerze-usługodawcy jest tworzona lub uaktualniana mapa (przez wydanie zlecenia *make passwd* w katalogu */var/yp*), która po rozpropagowaniu do pozostałych usługodawców w dziedzinie udostępnia klientom zawarte w niej informacje.

Rozpropagowanie tej samej informacji o danym użytkowniku za pomocą mapy NIS-owej odpowiadającej zbiorowi */etc/passwd* do wszystkich komputerów w dziedzinie nie tylko pozwala na „logowanie" się temu użytkownikowi do tych komputerów, ale zapewnia tworzenie (niezależnie od tego, na jakim komputerze w sieci) przez niego nowych zbiorów z takim samym identyfikatorem użytkownika i grupy (UId – *User Identyfication* i GId – *Group Identyfication*). Dzięki temu zbiory w całej sieci są związane z tym użytkownikiem, który je stworzył. Jest to szczególnie przydatne przy korzystaniu z sieciowego systemu dostępu do rozproszonych zbiorów znanym pod nazwą NFS – *Network File System*.

Lista standardowych map tworzonych i uaktualnianych przez usługę sieciową NIS na podstawie zbiorów zawartych w katalogu */etc* nie ogranicza się tylko do */etc/hosts* czy */etc/passwd*, jest znacznie większa (patrz wydruk).

bootparams	*mail.byaddr*	*passwd.byname*
ethers.byaddr	*netgroup.byhost*	*passwd.byuid*
ethers.byname	*netgroup.byuser*	*protocols.byname*
group.bygid	*netgroup*	*protocols.bynumber*
group.byname	*netid.byname*	*publickey.byname*
hosts.byaddr	*netmasks.byaddr*	*rpc.bynumber*
hosts.byname	*networks.byaddr*	*services.byname*
mail.aliases	*networks.byname*	*ypservers*

Można także sprawić, aby usługa NIS rozszerzyła swój mechanizm tworzenia, modyfikowania i udostępniania map na niestandardowe zbiory, tworząc z nich niestandardowe mapy (czyli takie, które nie zostały powyżej wymienione) oraz stając się tym samym usługą umożliwiającą zarządzanie rozproszonymi danymi w sieciach..

Na każdym komputerze włączonym do usługi NIS użytkownicy, a szczególnie administratorzy sieci mogą uzyskiwać informacje dotyczące jej aktualnego działania. Wydanie zalecenia systemowego *ypwhich* z różnymi opcjami, informuje użytkownika o wszystkich dostępnych mapach, udostępnianych przez określonych usługodawców (z opcją *-m*) lub o tym, który z usługodawców udzielą informacji zawartej w wyszczególnionej mapie (z opcją *-m⟨nazwa mapy⟩*). Wydanie z kolei zlecenia *ypcat⟨nazwa* mapy informuje użytkownika o zawartości danej mapy, przy czym lista takich zleceń i opcji nie ogranicza się do wyżej wymienionych.

* * *

Formułując wnioski dotyczące instalacji, działania oraz wykorzystania sieciowej usługi informacyjnej NIS, można stwierdzić, że:

- charakteryzuje się ona prostotą – proces instalacji i utrzymywania spójności map wymaga od administratora znajomości kilku lub kilkunastu poleceń systemowych;
- jest to usługa niezawodna – koncepcja hierarchicznej struktury usługodawców z podziałem na nadrzędnego i podrzędnego, a także baza protokołów, realizujących bezbłędną transmisję, zapewnia zgodność map NIS-owych w całej sieci;
- istniejąca możliwość ustanowienia wielu usługodawców w dziedzinie zmniejsza czas oczekiwania na uzyskanie dostępu do map przez wielu klientów w tym samym momencie;
- jest to usługa elastyczna pod względem konfigurowania; to, które z komputerów będą pełniły funkcje usługodawców zależy od administratora; jedynym wymaganiem jest, aby komputer usługodawcy miał masową pamięć dyskową, w której mógłby przechowywać mapy;
- istnieje możliwość rozszerzenia wcześniej określonych dziedzin przez dołączanie dodatkowych komputerów – usługodawców podrzędnych oraz klientów;
- nie ma żadnych ograniczeń, co do ustanawiania w sieci dziedzin NIS-owych; sugeruje się rozwiązania, aby w skład dziedziny wchodziły komputery, na których pracują użytkowni-

dokończenie na s. 24

JADWIGA ORYLSKA
Zakład Informatyki
Akademia Rolnicza
w Szczecinie

Dydaktyka informatyki w uczelniach rolniczych

Celem artykułu jest próba podzielenia się osobistymi spostrzeżeniami prowadzenia dydaktyki informatyki w uczelniach rolniczych, na przykładzie Akademii Rolniczej w Szczecinie. Masowa produkcja coraz lepszego sprzętu komputerowego powoduje upowszechnienie zastosowań metod i środków informatyki w różnych dziedzinach nauki i techniki, w tym również w gospodarce żywnościowej. Wywołuje to zmiany w procesach komunikowania się, gromadzenia, przechowywania, przetwarzania, przesyłania i udostępniania informacji. Obecnie każdy absolwent Akademii Rolniczej powinien dysponować wiedzą ogólną z dziedziny informatyki, stanowiącą nie tylko element wykształcenia ogólnego, ale również umożliwiającą aktywne korzystanie z jej osiągnięć. Oznacza to, że kształcenie na poziomie wyższym powinno zapewnić absolwentowi każdego kierunku studiów rolniczych (i nie tylko) opanowanie niezbędnej wiedzy i umiejętności, które pozwolą na wykorzystanie metod i środków informatyki w przyswajaniu wiedzy przedmiotów zawodowych oraz w pracy po skończeniu studiów. Jest to szczególnie konieczne w procesie obecnych przekształceń strukturalnych kraju. O potrzebie podnoszenia wiedzy informatycznej studentów dzisiaj nie trzeba już przekonywać. Problem jest natomiast w tym, jak to robić to dobrze i efektywnie.

Wielopoziomowość nauczania informatyki

Zarówno nauka matematyki jak i podstaw informatyki na studiach rolniczych rozpoczyna się równocześnie [1]. Student pierwszego roku ma oczywiście pewne przygotowania matematyczne, ale z nielicznymi wyjątkami – żadnego przygotowania informatycznego. Wykazuje natomiast bardzo duże zainteresowanie informatyką i oczekuje, że właśnie uczelnia dostarczy mu wiedzy na ten temat. Doznaje on bolesnego zawodu, ponieważ wiele uczelni jeszcze nie zawsze docenia znaczenie tego przedmiotu, co wyraża się zarówno w zbyt małej liczbie godzin, jak i w lokalizacji zajęć oraz wyposażeniu sal laboratoryjnych. Zwłaszcza na uczelniach rolniczych przedmiot ten jest traktowany marginesowo. Wykazała to dyskusja na II Ogólnokrajowej konferencji metodyki nauczania przedmiotów informatycznych, jaka niedawno odbyła się w Szczecinie. Potrzeba informatyzowania dydaktyki oraz konieczność jej nauczania bardzo wolno dociera do decydentów, co powoduje, że wszelkie decyzje w tej dziedzinie są najczęściej opóźnione ze szkodą dla przyszłych absolwentów. Uczelnia rolnicza naucza jedynie podstaw informatyki, a tylko czasem trochę więcej.

W trakcie kursu podstawowego student poznaje istotę działania komputera na przykładzie standardu IBM PC, elementy systemu operacyjnego, sens algorytmizacji oraz podstawy jednego z wybranych języków wysokiego poziomu. Zasadą jest równoległe poznawanie jednego z edytorów tekstu. Druga faza nauki obejmuje arkusz kalkulacyjny Lotus 1–2–3 oraz system relacyjnych baz danych, na przykładzie dBase III Plus. W przypadku wariantu jednosemestrowego, ogólny zakres przedmiotu nie zmienia się, ale często są gubione istotne szczegóły. Taki zakres tematyki pozwala zaledwie na korzystanie z gotowych narzędzi. Produkowany jest więc potencjalny klient informatyków, a nie ich konkurent wyposażony w dodatkową wiedzę specjalistyczną na temat przedmiotu zastosowania komputera. A właśnie specjaliści obu dziedzin tj. informatyki oraz zastosowania tworzą dziś postęp. Jednych i drugich obecnie nie brakuje [1]. Istnieje zatem konieczność, aby poza programem podstawowym, który powinien być nauczany w szkołach podstawowych i średnich (mam nadzieję, że w niedługim czasie tak będzie) wprowadzić zaawansowany poziom nauczania informatyki, który najpierw będzie obejmował zintegrowane oprogramowanie narzędziowe, a następnie nauczanie przedmiotów zawodowych w poszczególnych katedrach z wykorzystaniem techniki komputerowej. Ominięcie tego etapu, zwłaszcza na stanowiskach ekonomicznych, nie daje gwarancji odpowiedniego przygotowania absolwenta do pracy zawodowej. Będzie on niedouczony w posługiwaniu się narzędziami informatyki, z jednoczesnym bardzo specjalistycznym, tradycyjnym spojrzeniem na problem.

Co hamuje więc ten postęp? Obecnie istnieje wiele barier wdrażania informatyki w dydaktyce, takich jak:

- ograniczenie wymiaru godzin studiów wynikające z trudnej sytuacji ekonomicznej kraju, prowadzi do redukcji godzin zajęć z przedmiotu „Informatyka" i stwarza niebezpieczeństwo coraz większego niedopasowania sylwetki absolwentów uczelni rolniczych do nowych potrzeb rynku pracy,
- niedostatek sprzętu komputerowego (sprzęt niesprawny lub przestarzały),
- brak kwalifikowanej kadry informatyków; spotyka się pogląd, że informatyki może uczyć każdy, kto zna obsługę komputera oraz instrukcję obsługi określonego pakietu narzędziowego.

Cel i treść nauczania informatyki

Istotnym problemem, który wyraźnie stwierdzono w bezpośrednich sondażach ze studentami jest dostosowanie nauczania informatyki do sylwetki absolwenta studiów rolniczych na miarę obecnych potrzeb i oczekiwań praktyki gospodarczej. Oznacza to konieczność zbliżenia programu tego nauczania do praktyki krajów wysoko rozwiniętych, mimo istniejących barier sprzętowych, programowych i kadrowych, oraz maksymalne położenie nacisku na praktyczne posługiwanie się sprzętem i oprogramowaniem.

Cały proces kształcenia informatycznego należy podzielić na dwa poziomy: poziom I – podstawowy oraz poziom II – zaawansowany [2, 3, 4, 5].

Poziom I (podstawowy), realizowany na pierwszym lub drugim roku studiów obligatoryjnie w wymiarze 15 godzin wykładów i 45 godzin ćwiczeń, obejmuje następujące wykłady:
- podstawowe pojęcia informatyki,
- przegląd podstawowych kategorii sprzętu komputerowego i oprogramowania oraz pracy w sieci,
- podstawy działania systemów informatycznych w technologii i zarządzaniu gospodarką żywnościową,
- wstępne wiadomości z zakresu programowania komputerów i zasad ich użytkowania.

Ćwiczenia laboratoryjne prowadzone w grupach po 6–8 osób (1–2 studentów na jedno stanowisko komputerowe) obejmują:
- naukę obsługi komputera klasy IBM PC,
- umiejętność pracy pod nadzorem systemu operacyjnego MS-DOS z ewentualną możliwością wykorzystania nakładek,
- przyjazne dla studenta, oprogramowanie narzędziowe, np. edytor tekstów, arkusz kalkulacyjny, dBase lub zasady programowania w wybranym języku programowania, typu Turbo, BASIC czy TURBO PROLOG.

Celem tego poziomu kształcenia jest opanowanie umiejętności korzystania z narzędzi informatyki na tyle, aby w dalszych latach studiów student mógł korzystać z nich w nauczaniu przedmiotów ogólnych i zawodowych. Poziom ten ma zadanie stworzyć studentom możliwość dalszego pogłębienia wiadomości z informatyki.

Poziom II (zaawansowany) jest ukierunkowany na zastosowania i może być potraktowany jako fakultatywny. Byłby on przeznaczony dla studentów mających podstawy ze szkoły średniej lub studentów, zainteresowanych poszerzeniem wiadomości z informatyki na III, lub IV roku studiów. Nie dotyczy to studentów Wydziału Ekonomiki i Organizacji Gospodarki Żywnościowej, gdzie poziom ten powinien być traktowany obligatoryjnie.

Poziom zaawansowany obejmuje wszystkie zagadnienia poziomu podstawowego, ale nauczane w sposób bardziej szczegółowy. Obejmuje on na wszystkich wydziałach 15 godzin wykładów oraz 45 godzin ćwiczeń.

Wykłady obejmują:
- zintegrowane oprogramowanie narzędziowe,
- oprogramowanie specjalistyczne, dostosowane do sylwetki absolwenta,
- systemy informatyczne,
- pakiety inżynierskie i statystyczne,
- obliczenia optymalizacyjne, symulację komputerową, sztuczną inteligencję, systemy ekspertowe.

Ćwiczenia, prowadzone w grupach laboratoryjnych 6–8 osobowych (1–2 studentów na jedno stanowisko komputerowe) powinny obejmować:
- edytory tekstów,
- arkusze kalkulacyjne,
- dBase III Plus lub (IV),
- systemy informatyczne,
- pakiety programów statystycznych, optymalizacyjnych, inżynierskich,
- kalkulację kosztów,
- sztuczną inteligencję i systemy ekspertowe z wykorzystaniem bazy danych oraz bazy wiedzy.

Potrzeby kształcenia informatycznego na poszczególnych wydziałach byłyby realizowane fakultatywnie według zainteresowań studentów i koncentrowałyby się wokół następujących zagadnień [2, 3, 4, 5]:

Wydział Rolniczy – doradztwo płodozmianowe i odmianowe, optymalizacja struktury zasiewów, opłacalność poszczególnych kierunków produkcji, podstawowe wskaźniki produkcyjno-finansowe, struktura gruntów (przyrost, ubytek, klasy-analiza i prognoza), optymalizacja produkcji, warunki sprzedaży produkcji roślinnej i dochody (analiza i prognoza), wydajność zbóż, okopowych, strączkowych (analiza i prognoza), potrzeby paszowe produkcji zwierzęcej, rodzaj i terminarz stosowanych środków ochrony roślin, doradztwo nawozowe, zabiegi agrotechniczne, realizacja przebiegu prac polowych, skutki niedotrzymania terminów prac agrotechnicznych (nawożenia, oprysków, zbiorów), problemy inżynierskie.

Wydział Zootechniczny – rentowność i struktura hodowli oraz chowu zwierząt gospodarskich, podstawowe wskaźniki produkcyjno-finansowe, struktura i stan stada, urodzenia i upadki zwierząt, obrót stada, przyrosty wagowe, wydajność mleka, żywca, wełny, wielkość produkcji i sprzedaży, gospodarka paszowa (optymalizacja bazy paszowej), wpływ żywienia na produkcyjność zwierząt i skutki niedoborów paszowych, wpływ pochodzenia zwierząt na ich produkcyjność i kondycję, profilaktyka i leczenie zwierząt, problemy inżynierskie.

Oddział Techniki Rolniczej – eksploatacja sprzętu rolniczgo i jego opłacalność, wykorzystanie środków transportu (ograniczenie pustych przebiegów), diagnostyka i konserwacja sprzętu, problemy inżynierskie.

Wydział Rybactwa Morskiego i Technologii Żywności – ewidencja połowów (krajowa i światowa), przetwórstwo rybne (statystyka), ekonomika gospodarki rybnej w przedsiębiorstwach i w regionie, organizacja gospodarstw rybnych (ewidencja i obsługa informatyczna), problemy inżynierskie.

Wydział Ekonomiki i Organizacji Gospodarki Żywnościowej – finanse i księgowość, kalkulacja kosztów, kredyty, oprocentowanie, akcje, system handlu zagranicznego, marketing, zatrudnienie i płace, gospodarka materiałami i produktami rolniczymi, inwestycje i środki trwałe, opłacalność produkcji rolniczej, bank informacji gospodarki żywnościowej (informacje giełdowe, kursy walutowe itp.), problemy inżynierskie.

Powyższa problematyka powinna być rozpatrywana w odniesieniu do czasu przeszłego, teraźniejszego i przyszłego (ewidencja, analiza, plan).

Naukę zaawansowanego programowania w wybranym języku studenci mogą podjąć w ramach przedmiotów nieobowiązkowych lub też, poza programem informatyki, na specjalistycznych kursach organizowanych w Uczelni (lub poza nią) na zasadzie pełnej odpłatności.

Kontrola wiadomości

W kształceniu informatycznym studenci powinni sprawdzać swoje umiejętności przez samodzielne rozwiązywanie problemów. Zadania stawiane studentom dotyczą zwykle prostych zagadnień z gospodarki żywnościowej, a wynik otrzymany z arkusza kalkulacyjnego, edytora tekstów, czy bazy danych służy jedynie kontroli. Zbiór zadań kontrolnych powinien być urozmaicony i ciągle wzbogacany przez prowadzącego zajęcia. Bardzo istotną sprawą jest oprogramowanie specjalistyczne, realizowane na poziomie zaawansowanym w Zakładzie Informatyki i następnie w poszczególnych katedrach (w ramach zajęć specjalistycznych). Prowadzący zajęcia na ostatnich (specjalistycznych) latach studiów powinien na bieżąco dostosowywać programy zajęć do możliwości jakie stwarzają mikrokomputery. Nie ulega więc wątpliwości, że wykorzystanie mikrokompu-

terów jest szczególnie istotne w zakresie zajęć typu ćwiczeniowego, seminaryjnego i projektowego. Niewielkie grupy studenckie oraz dysponowanie mikrokomputerami przez poszczególne katedry przenosi ciężar dydaktyki na lata starsze, gdzie następuje praktyczne wykorzystanie wiedzy nabytej na zajęciach z przedmiotu „Informatyka". Tu nauka powinna dotyczyć umiejętności posługiwania się profesjonalnymi programami specjalistycznymi. Dobre przygotowanie studenta, to nie tylko zaliczenie przez niego przedmiotu „Informatyka", ale także dydaktyka informatyzacji przedmiotów specjalistycznych, dostosowanych do nowych warunków ekonomicznych kraju. Realizacja podanego procesu powinna opierać się na komputerach nie tylko klasy IBM PC/AT, ale również na sprzęcie o lepszych parametrach technologicznych. Należy również wykorzystywać nie tylko standardowe oprogramowanie użytkowe, ale również sieć akademicką (NASK).

* * *

Z przeprowadzonych przez autorkę badań wynika, że dydaktykę informatyki należy oprzeć na określonej, konsekwentnie realizowanej koncepcji informatyzacji całego procesu dydaktycznego Uczelni. Zakłada ona wiele przedsięwzięć, z których do najważniejszych zaliczyć można:
- przygotowanie całej kadry dydaktycznej Akademii Rolniczej do stosowania metod i środków informatyki w stopniu podstawowym; realizować to można przez organizowanie kursów komputerowych oraz wydawanie periodyków (np. biuletyn „Informatyka w dydaktyce", redagowany głównie przez pracowników Zakładu Informatyki AR-Szczecin);
- opracowanie scenariuszy i komputerowych programów dydaktycznych ukierunkowanych na gospodarkę żywnościową (produkcję roślinną, zwierzęcą, przetwórstwo);
- systematyczne wprowadzanie do podstawowych programów nauczania nowoczesnych metod i środków informatyki, a na poziomie zaawansowanym – także specjalistycznej informatyki rolniczej; przedmiot ten od dwóch lat jest realizowany na dziewiątym semestrze w SGGW-AR-Warszawa, wymiennie z przedmiotem ekologia, który cieszy się mniejszym powodzeniem; sądzę, że omawiany przedmiot można by wprowadzić również w naszej Uczelni.

Przy tak realizowanej koncepcji dydaktyki student Akademii Rolniczej w trakcie studiów zetknie się z informatyką jako:
- przedmiotem nauczania,
- narzędziem warsztatowym studenta, a w przyszłości inżyniera rolnictwa, rybactwa czy ekonomiki,
- narzędziem dydaktycznym wspomagającym proces nauczania wraz z kontrolą i oceną materiału dydaktycznego.

Absolwent opuszczając Akademię Rolniczą powinien mieć pełną świadomość możliwości wykorzystania metod, technik i narzędzi informatyki, jako obecnie niezbędnego nowoczesnego warsztatu inżynierskiego na każdym stanowisku pracy.

LITERATURA

[1] Dytczak M.: Wielopoziomowe nauczanie przedmiotów informatycznych w uczelniach rolniczych. MOM SGGW-AR – Warszawa i AR – Szczecin, 1991, s. 20–29 (materiały konferencyjne)
[2] Orylska J.: Informatyka jako przedmiot nauczania w akademiach rolniczych na przykładzie AR – Szczecin. MOM SGGW-AR – Warszawa, 1991, s. 18–25 (materiały konferencyjne)
[3] Orylska J.: Z badań nad modernizacją systemu nauczania informatyki w akademiach rolniczych (na przykładzie Akademii Rolniczej w Szczecinie). MOM SGGW-AR – Warszawa, Studia, materiały, informacje, zesz. 2/59, 1991, s. 32–34
[4] Orylska J.: Umiejętności absolwenta Akademii Rolniczej w zakresie informatyki w XXI wieku. Sylwetka absolwenta Akademii Rolniczej XXI wieku. AR – Szczecin, PAN – Oddział Gdańsk, 1991 r., s. 55–59
[5] Orylska J.: Nauczanie informatyki w nowych warunkach ekonomicznych kraju. Metodyka nauczania przedmiotów informatycznych w uczelniach rolniczych. MOM SGGW-AR – Warszawa. ZI AR – Szczecin, 1991, s. 10–19 (materiały konferencyjne)
[6] Wesołowska-Janczarek M.: Przedmiot „Informatyka" i kilka problemów związanych z jego realizacją. Metodyka nauczania przedmiotów informatycznych w uczelniach rolniczych. MOM SGGW-AR – Warszawa i ZI AR – Szczecin, 1991, s. 30–34.

Dlaczego NIS?

dokończenie ze s. 21

cy mający wspólnie określone cele pracy lub zajmujący określony wspólny obszar (biuro, pokój, piętro, budynek);
- dzięki możliwości tworzenia, uaktualniania i udostępniania do odczytu niestandardowych map, usługa NIS staje się systemem rozproszonych baz danych w UNIX-ie;
- z uwagi na bazę protokołów, na których opiera się działanie usługi NIS (przede wszystkim tych z rodziny TCP/IP), może ona pracować na różnych komputerach, na których są zainstalowane różne implementacje systemów operacyjnych UNIX (dodajmy, że dzięki takiemu produktowi, jak pakiet programowy PC-NFS firmy Sun Microsystems, usługa NIS jest zaimplementowana również w systemie DOS).

Niniejszy artyku jest wynikiem prac prowadzonych w Zespole Heterogenicznych Systemów Komputerowych w Instytucie Informatyki Teoretycznej i Stosowanej PAN w Gliwicach. Sieciowa usługa informacyjna NIS została uruchomiona na lokalnej, ethernetowskiej sieci na bazie protokołów TCP/IP, do której były przyłączone następujące komputery: SUN SPARC station 1+, SUN SPARC station IPC (pracujące pod systemem operacyjnym SunOS 4.1.1) oraz VAX station II/RC (pracujący pod systemem operacyjnym ULTRIX-32), natomiast po zainstalowaniu pakietu PC-NFS, również na mikrokomputerach Business VEISA ALR 386 pracujące pod systemem DOS (wersja 3.1 lub nowsza); te ostatnie mogły być tylko usługobiorcami usługi NIS. W systemach operacyjnych SunOS 4.1.1 i ULTRIX-32 elementy wchodzące w skład usługi NIS są integralną częścią tych systemów.

LITERATURA

[1] Sun Microsystems (SunOS 4.1.1) – System & Network Administration, Revision A of 27 March 1990
[2] ULTRIX-32 (Version 2.0) – Network Management Guide, 1987
[3] UNIX – Światowy Standard Wielodostępu (rozdział: UNIX w sieciach, autor: Jacek Surma), FRTK Warszawa 1991.

Przypominamy!

Jedyną możliwością regularnego otrzymywania INFORMATYKI jest prenumerata.

Czasopismo można zaprenumerować w dowolnym czasie i na dowolny okres w Dziale Kolportażu Wydawnictwa SIGMA-NOT Spółka z o.o. **00-716 Warszawa, ul. Bartycka 20, skr. poczt. 1004, telefony: 40-30-86, 40-35-89 lub 40-00-21 w. 293, 295, 299.**

Zaległe numery można zawsze kupić w redakcji **tel. 39-14-34** lub **Dziale Handlowym: 00-950 Warszawa, skr. poczt. 1004, tel. 40-37-31.**

TEODOR WINKLER
Pracownia Zastosowań Komputerów
w Projektowaniu i Wytwarzaniu
Katowice

Środowisko systemów CAD

Jeszcze do niedawna systemy CAD (*Computer Aided Design*) były traktowane jako samodzielne i całkowicie autonomiczne jednostki programowe, opisujące właściwości geometryczne tworzonych obiektów. Przedmiotem zainteresowania było przede wszystkim tworzenie dokumentacji rysunkowej w jej dotychczasowej formie. Rysunki w rzutach prostokątnych stanowiły główny rezultat działań. Modelowanie geometryczne otworzyło możliwości pełnego zapisu cech geometrycznych obiektów i dało podstawy pod drugi składnik tego, co określa się mianem systemów CAD–CAM.

Wspomaganie komputerowe projektowania technologii, a w szczególności tej jej części, która jest realizowana na obrabiarkach numerycznie sterowanych, łączy się bezpośrednio z wytwarzaniem. Rozwój metod wytwarzania w kierunku tzw. elastycznych systemów wytwarzania zwrócił uwagę na konieczność integracji procesów projektowo-konstrukcyjnych oraz wytwarzania. Tak powstało pojęcie CIM (*Computer Integrated Manufacturing*), czyli komputerowo zintegrowanego wytwarzania, do którego zalicza się nie tylko tradycyjnie rozumiane wytwarzanie, lecz również wszystkie te procesy, które się z nim wiążą.

Wysiłkom integracyjnym w dziedzinie wytwarzania towarzyszą podobne wysiłki w obrębie projektowania i konstruowania. Dwie podstawowe, i jak dotąd odrębnie traktowane grupy czynności, a mianowicie obliczenia inżynierskie i sporządzanie rysunków technicznych, są coraz ściślej ze sobą wiązane. Metoda elementu skończonego stanowi tu przykład już prawie klasyczny. Dostęp do graficznej bazy danych systemów CAD pozwala rozciągnąć integrację również na inne zagadnienia, jak np. analizę kinematyczną i analizę statyczną. Szczególnie skuteczne okazują się tu metody modelowania graficznego.

W tradycyjnej organizacji wytwórni procesy bezpośrednio poprzedzające wytwarzanie noszą nazwę technicznego przygotowania produkcji. Na podstawie dokumentacji konstrukcyjnej są tworzone dokumenty bezpośrednio opisujące procedury wytwarzania oraz pozwalające zorganizować zaopatrzenie, zbyt i gospodarkę magazynową. Systemy zajmujące się tą częścią wytwarzania noszą nazwę systemów PPS (*Product Planing System*).

Bliższe badania wykazują, że koszty wytworu w 70% są określone już w trakcie wczesnych faz procesu konstrukcyjnego. Szacuje się, że fazy te obejmują 15% czasu opracowania nowej konstrukcji. Szybka kalkulacja kosztów wytwarzania, oparta na danych pochodzących z systemu CAD, pozwala na analizę wielu wariantów konstrukcjnych w świetle kryteriów ekonomicznych.

Modelowanie graficzne a parametryzacja konstrukcji

Parametryzacja konstrukcji w powszechnym rozumieniu odnosi się przede wszystkim do parametryzacji wymiarów dla określonej postaci konstrukcyjnej. Postać konstrukcyjna jest przy tym traktowana jako pewna całość składająca się z określonej liczby elementów o ustalonym wzajemnym położeniu. Mówiąc inaczej, jest to stała postać konstrukcyjna i zmienny układ wymiarów. W praktyce, pojęcie zmienności obejmuje często również wzajemne usytuowanie poszczególnych elementów.

Bliższego wyjaśnienia wymaga w tym miejscu pojęcie elementu. Konstruktor w swojej pracy posługuje się elementami, które mają znaczenie nośników o ich materialnych odpowiednikach. Mogą to być informacje mające znaczenie przede wszystkim geometryczne (np. rowki, podcięcia). Takie elementy nazwać można elementami postaciowymi i z nich przede wszystkim tworzy się rysunki warsztatowe. Użyte do ich zapisu elementy geometryczne wiernie odwzorowują obiekty rzeczywiste.

Stosowane dotychczas narzędzia (np. PARA–CAD, GLISP) odnoszą się głównie do parametryzacji postaci konstrukcyjnej pojedynczych elementów maszyn. Jeżeli z elementem jest związana ponadto informacja o jego działaniu, to z uwagi na zmniejszenie złożoności opisu użyte tu elementy geometryczne mają znaczenie symbolu. Informacje niesione przez sam symbol są dopełnione przez dodatkowe atrybuty. Stosowane są liczne biblioteki symboli z dziedziny kinematyki, hydrauliki, elektrotechniki. Ich dotychczasowe znaczenie polegało przede wszystkim na uproszczonym zapisie zasady działania środków technicznych. Te same biblioteki w środowisku systemów CAD spełniają swoją dotychczasową rolę, a ponadto utworzone z nich zapisy mogą być przedmiotem wielorakich analiz.

Metoda modelowania graficznego przewiduje zbiór elementów graficznych (symboli) reprezentujących zarówno elementy materialne, jak i symbole wyobrażające stan obciążenia. W niektórych przypadkach nie jest wymagane graficzne przedstawienie stanu obciążenia, gdyż wynika on wprost ze wzajemnego położenia względem siebie współpracujących elementów materialnych. Przekładnie zębate należą do tych konstrukcji, które trudno poddają się parametryzacji, pomimo że dobór cech konstrukcyjnych poszczególnych elementów: kół, wałków, łożysk przebiega według dobrze znanych zasad. Główną trudność sprawia duża różnorodność w rozmieszczeniu kół, różne usytuowanie wałków wejściowych i wyjściowych, zmieniająca się liczba stopni. Te oraz inne czynniki utrudniają opis zmienności postaci konstrukcyjnej przekładni zębatych.

Pakiet AutoGEAR wykorzystujący metody modelowania graficznego pozwala na skuteczne uchwycenie tej zmienności dla praktycznie nieograniczonej liczby wariantów przekładni zębatych o zazębieniu czołowym. Jest on całkowicie zintegrowany z systemem AutoCAD.

Pakiet jest sterowany własnym menu rozwijanym, którego opcje wspomagają takie czynności, jak: sporządzanie schematu kinematycznego i jego edycję, dobór cech konstrukcyjnych kół, analizę kinematyczną, obliczenia wytrzymałościowe, identyfikację stanu obciążenia i inne. Odpowiednie mechanizmy usprawniają i przyspieszają tworzenie schematu kinematycznego.

Symbole graficzne potrzebne do sporządzenia schematów kinematycznych są umieszczone w specjalnie do tego celu przeznaczonym menu obrazkowym. Sposób, w jaki symbole te są rozmieszczane na ekranie monitora, nie odbiega od szkicowania na kartce papieru. Modelowane łańcuchy kinematyczne mogą składać się z kół z zębami prostymi lub skośnymi. Mogą się tam znaleźć również przekładnie planetarne. Każde koło, wałek i łożysko jest opatrzone własnym numerem pozycji.

Wprost ze schematu kinematycznego jest rozpoznawana droga przeniesienia mocy z wału wejściowego na wał wyjściowy. Identyfikowane są stan obciążenia przekładni oraz prędkości obrotowe wszystkich elementów wirujących. Obliczenia geometryczne i wytrzymałościowe są prowadzone bądź to dla wszystkich kół, bądź dla wskazanej pary kół.

Obliczone cechy konstrukcyjne pozwalają sporządzać rysunki kół bez potrzeby korzystania z norm i bez dodatkowych obliczeń.

Model graficzny służy ponadto do rozwiązywania różnych szczegółowych zadań. Mogą to być obliczenia: odległości międzyosiowej pary kół, wymaganej sumy korekcji, masowych momentów bezwładności kół, przybliżonej wartości ciężaru wszystkich kół itd.

Model graficzny zawiera wystarczające informacje, by utworzyć uproszczony rysunek złożeniowy przekładni. W tym celu model graficzny zostaje przekształcony w model geometryczny. Możliwe jest przedstawienie przekładni zarówno w postaci rysunku przestrzennego, jak i w postaci rysunków w rzutach prostokątnych.

Jednolite biblioteki elementów znormalizowanych

Stosowanie elementów znormalizowanych daje duże efekty racjonalizacyjne. Bliższe badania wskazują, że nawet nowe konstrukcje w 50% zawierają elementy znormalizowane.

Biblioteki elementów znormalizowanych są stosowane już od dawna w systemach CAD. Były one tworzone dla konkretnych systemów i nie zawsze obejmowały całość norm. Ograniczało to przenośność bibliotek między systemami oraz utrudniało pracę użytkowników stosujących kilka systemów. Z inicjatywy głównej niemieckiej instytucji normalizacyjnej DIN (*Deutsches Institut für Normung*) powstał projekt jednolitej biblioteki elementów znormalizowanych, niezależnej od konkretnego systemu CAD. Wiąże się to z opracowaniem zasad tworzenia konstrukcji wariantowej, bo tym w istocie są elementy znormalizowane. Rezultatem tych starań są normy DIN V 4001 i DIN V 666304. Ta ostatnia jest znana również pod nazwą VDAPS (*Verband der Automobilindustrie-Programmschnittstelle*). Norma DIN V 4001 zawiera jednolite zasady tworzenia zapisu cech konstrukcyjnych elementów znormalizowanych. Tradycyjne arkusze norm zostają przekształcone w tabele wymiarów o odpowiednim formacie. Tabele te następnie są wczytywane przez programy generujące kompletny zapis konstrukcji zgodny z normą VDAPS. Programy te, napisane w języku FORTRAN 77, tworzą poszczególne warianty elementów znormalizowanych. Norma ta jest jednocześnie podstawą do tworzenia interfejsów łączących biblioteki z różnymi systemami CAD. W podobny sposób powstają biblioteki norm zakładowych czy biblioteki tzw. elementów kupnych.

Użytkownik dysponujący interfejsem VDAPS może korzystać z każdej utworzonej zgodnie z przedstawionymi zasadami, biblioteki elementów znormalizowanych. W połowie 1991 r. biblioteki te obejmowały już ok. 900 norm. Hewlett-Packard zastosował normę DIN tworząc interfejs HP VDA–PS do własnych systemów ME10 i ME30. Obsługę bibliotek elementów znormalizowanych wspomaga specjalny moduł uwzględniający logikę konstruowania z użyciem tych elementów. I tak np. jest wyliczana wymagana długość części złącznych na podstawie wskazanych na ekranie punktów leżących na powierzchniach łączonych. W złożeniach, jak np. śruba-nakrętka-podkładka, w sposób automatyczny są wygaszane linie zakryte. Informacje o zastosowanych elementach są przekazywane do generatora wykazów elementów. Podobne rozwiązanie zostało przyjęte w systemach CADAM (CADAM Inc.) i CATIA (*Dassault Systèmes*). Pakiet zarządzający bibliotekami elementów znormalizowanych o nazwie FAST z niemieckiej firmy DVO-Datenverarbeitungs-Service Oberhausen GmbH jest wyposażony w język opisu logiki konstruowania CMS (*Component Management System*). Kreśleniem elementów oraz wygaszaniem linii niewidocznych steruje moduł VLM (*View Logic Module*).

Systemy CAD a systemy przygotowania produkcji

Systemy przygotowania produkcji stanowią, jak dotąd, odrębną dziedzinę, której przedmiotem jest przetwarzanie informacji dla potrzeb produkcji, zaopatrzenia i zbytu. W szczególnym przypadku mogą to być informacje o geometrycznych cechach konstrukcyjnych. Zazwyczaj jednak systemy te korzystają z danych zawartych w wykazach elementów. Towarzyszą one dokumentacji rysunkowej i stanowią odwzorowanie struktury wytworu, czyli podziału na zespoły, podzespoły i elementy. Z kolei systemy PPS gromadzą dane potrzebne w czasie procesu projektowego, np. dotyczące cen, ciężarów elementów kupnych, pracochłonności itd. Bazy danych zarządzane systemami mogą być analizowane np. pod kątem częstości występowania niektórych elementów lub ich podobieństwa technologicznego. Istnieją dwa podstawowe sposoby łączenia tych systemów z systemami CAD:

- Przekazywanie zbiorów utworzonych w systemie CAD do komputera pracującego w systemie PPS. Dane są przechowywane w dwóch oddzielnych bazach i są porównywane ze sobą w określonych odstępach czasu. Tylko w tym momencie zawartość obydwu baz jest identyczna. Konstruktor nie ma bezpośredniego dostępu do danych przechowywanych w zbiorach PPS;
- Systemy CAD i PPS są ze sobą zintegrowane. Konstruktor przekazuje wykazy elementów bezpośrednio do systemu PPS. Wyniki działania systemu PPS są natychmiast widoczne na monitorze systemu CAD. Dane są przechowywane tylko w jednej bazie.

Taki sposób rozwiązania, to propozycja szwajcarskiej firmy INWECO CAD-System AG i dotyczy projektów realizowanych w systemie Auto-CAD.

Rysunkowa baza danych jest zarządzana pakietem AutoZv działającym pod kontrolą systemu MS–DOS lub PC–DOS. Każdy rysunek jest opisany przez numer rysunku (do 20 pozycji), nazwę, materiał, wymiary gabarytowe, format, podziałkę i przez dane organizacyjne. Rysunki są wyszukiwane według każdej z podanych cech opisujących rysunek. Opisy tekstowe tych cech mogą być w formie pełnej lub skróconej.

Z rysunkami opisującymi rysunek jest związane pole, w którym są umieszczone dodatkowe, bliższe opisy słowne. W specjalnym bloku rekordów są umieszczone teksty dotyczące zmian na rysunku. W tej samej bazie jest przechowywana struktura projektu uwzględniająca podział do 7 stopni hierarchii w głąb. Odwzorowuje ona powiązania między rysunkami złożeniowymi całości, zespołów i podzespołów, aż do poziomu elementów. Wydruk struktury projektu po zmodyfikowaniu może być przekształcony w wykaz rysunków z dokładnym określeniem numerów rysunków, nazw, dat itd. Użytkownik komunikuje się z pakietem przez plansze ukazujące się na tle aktualnie tworzonego rysunku.

Częścią proponowanego rozwiązania jest generator wykazów o nazwie *AutoBase*. Generator ten komunikuje się z bibiotekami elementów znormalizowanych i umieszcza je w aktualnie tworzonych wykazach elementów. Użytkownik nadaje numery poszczególnym pozycjom na rysunku, przy czym generator rozpoznaje nadane wcześniej numery i dokonuje automatycznego przenumerowania. Istnieje bezpośrednia komunikacja z plikami DWG z pominięciem drogi przez pliki DXF. Zarejestrowane numery pozycji zostają umieszczone w bazie danych.

Wygenerowane wykazy elementów mogą podlegać edycji. Wybór wykazu podlegającego edycji następuje wprost spod AutoCAD-a za pomocą specjalnej planszy.

Istnieje ponadto możliwość tworzenia wykazów uporządkowanych według różnych kryteriów sortowania. Może to być m.in. wykaz cen wszystkich zastosowanych w danym projekcie elementów znormalizowanych, zestawienie numerów pozycji, wykaz cen wszystkich składników projektu. Zwraca się uwagę na łatwą wymianę danych z systemami PPS.

Przykładem całkowitej integracji systemów CAD i PPS są systemy KONSYS r. 7 i PSK 200 firmy Strässle GmbH. System PSK 200 pełni rolę integrującą w ramach koncepcji CIM. Z tego powodu funkcje realizowane przez system KONSYS są zaliczane do funkcji planowania, przypisanej systemowi PSK 200. Jednocześnie istnieje pełna autonomia obydwu systemów. Wspomagane są wszystkie fazy rozwojowe wytworu – od sporządzenia oferty, przez gospodarkę materiałową, wytwarzanie, wysyłkę aż po fakturowanie. System PSK 200 jest oparty na relacyjnej bazie danych Oracle i eksploatowany na stacjach roboczych HP 9000 pod nadzorem systemu HP–UX oraz na IBM RISC 6000 poda nadzorem systemu AIX. System jest przeznaczony dla przedsiębiorstw średniej wielkości i ma strukturę modułową: zarządzanie, zbyt, wykazy elementów, gospodarka materiałowa i zaopatrzenie, harmonogramowanie produkcji.

Użytkownik systemu CAD ma bezpośredni dostęp do bazy danych systemu PPS.

Komputerowo wspomagana kontrola jakości CAQ

W łańcuchu działań objętych terminem CIM znajduje się również kontrola jakości CAQ (*Computer Aided Quality Assurance*). Obecny stan techniki pomiarowej pozwala włączać modele utworzone w systemach CAD do procesów pomiarowych, których przedmiotem są obiekty rzeczywiste. Model komputerowy jest tu wzorcem, modelem odniesienia. Na maszynie pomiarowej następuje digitalizacja obiektu rzeczywistego (do 5 osi), sterowana procesorem wbudowanym w komputer maszyny pomiarowej. Uzyskane w ten sposób dane są przekazywane do systemu CAD. Wymiana danych o charakterze graficznym przebiega według jednej ze wspomnianych norm, np. VDAPS.

Rysunki warsztatowe z systemu CAD, zawierający wymiary, oznaczenia chropowatości i tolerancje, są przekazywane do tzw. modułu *Q*, w którym jest tworzony program pomiarów realizowany na maszynie pomiarowej. Uzyskane wyniki pomiarów stanowią sprzężenie zwrotne dla systemu CAD. Eliminuje to żmudne i pracochłonne tworzenie programów pomiarowych metodą *teach-in* bezpośrednio na maszynie pomiarowej.

* * *

Współczesne systemy CAD pracują w szerokim środowisku programowym wspomagającym procesy towarzyszące projektowaniu. Środowisko to znajduje się w ustawicznym rozwoju. Zmiany w sprzęcie wymuszają zmiany w oprogramowaniu. Przykładem tego są komputery oparte na technologii RISC. Sama dziedzina wspomagania komputerowego liczy już sobie około 20 lat, można zatem mówić o pewnych zaszłościach historycznych znajdujących swoje odbicie w eksploatowanych u użytkowników sprzęcie i oprogramowaniu. O ile jeszcze do niedawna można było forsować określone standardy związane z konkretnym producentem oprogramowania, o tyle dzisiaj potencjalny użytkownik CAD może wybierać rozwiązania oparte na standardowych nakierowaniach na rozwiązywanie szczegółowych zadań: wymiana danych między systemami, parametryzacja, biblioteki specjalistyczne, programy – aplikacyjne. Coraz częstsza jest sytuacja, że ten sam użytkownik ma u siebie kilka systemów i jest zmuszony współpracować z kooperantem pracującym na jeszcze innym systemie.

Tworzy się więc pewien zbiór warunków brzegowych, które muszą brać pod uwagę firmy już istniejące na rynku polskim, jak również i te, które na ten rynek chcą dopiero wejść.

Wskazówki dla Autorów

Nadsyłane artykuły nie mogą być publikowane lub przeznaczone do opublikowania w innych czasopismach.

Materiał oprócz tekstu zasadniczego powinien zawierać:

- krótki życiorys zawodowy Autora i jego zdjęcie,
- wykaz literatury,
- tabele,
- materiał ilustracyjny (rysunki, zdjęcia czarno-białe, wydruki) dołączony do artykułu (nie wklejać materiału w tekst),
- podpisy pod ilustracje.

Na osobnej stronie prosimy podać: tytuł naukowy, nazwisko i imię, nazwę zakładu pracy, numer telefonu służbowego oraz informację, jaką drogą przekazać honorarium – kasa Wydawnictwa, poczta, konto.

W związku z naliczaniem podatku dochodowego prosimy także podać:

- oprócz nazwiska i imienia – także drugie imię,
- imiona rodziców,
- datę i miejce urodzenia,
- numer identyfikacyjny PESEL,
- miejsce zameldowania,
- adres Urzędu Skarbowego właściwego dla miejsca zamieszkania Autora.

Tekst artykułu prosimy dostarczyć w formie maszynopisu lub wydruku komputerowego, pisanego jednostronnie, z podwójnym odstępem (30 wierszy na stronie i 60 znaków w wierszu).

Rysunki powinny być czytelne i mogą być wykonane ołówkiem.

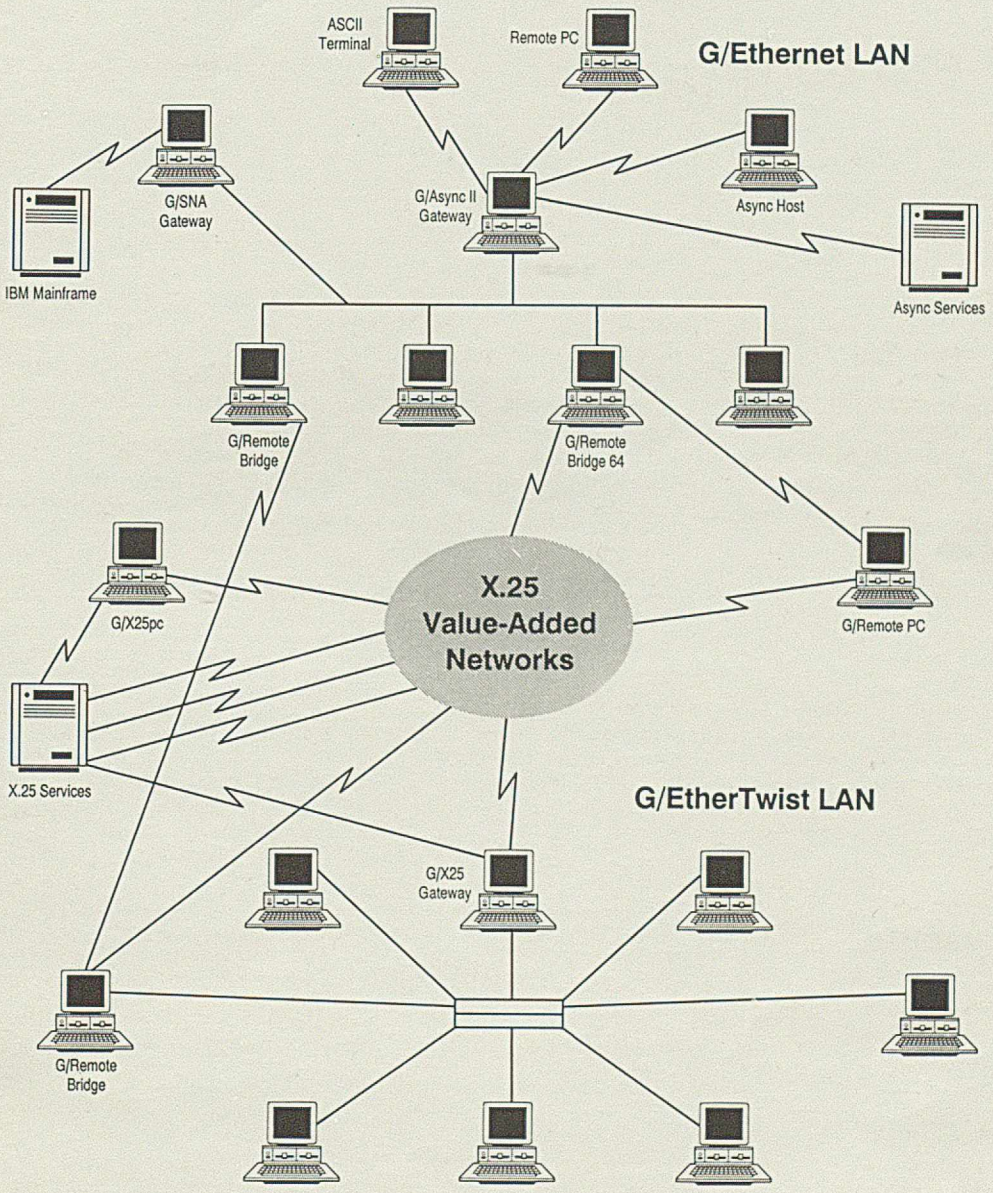
Autor opublikowanego artykułu otrzymuje honorarium i bezpłatny egzemplarz okazowy.

Materiałów nie zamówionych redakcja nie zwraca.

Stabrowski M.: AMIL – interakcyjny język symulacji układów dynamicznych INFORMATYKA 1992, nr 11, s. 1 Charakterystyka języka programowania AMIL oraz jego porównanie z innymi współczesnymi językami do symulacji układów dynamicznych.	**Stabrowski M.: AMIL – an interactive language for dynamic circuits simulation** INFORMATYKA 1992, No. 11, p.1 Characteristics of the AMIL programming language and its comparison with other contemporary languages for dynamic circuits simulation.	**Stabrowski M.: AMIL – eine interaktive Sprache für dynamische Simulation der Schaltungen** INFORMATYKA 1992, Nr.. 11, S. 1 Eine Charakteristik der AMIL-Programmiersprache und ihrer Vergleich mit anderen zeitgenössischen Sprachen für dynamische Simulation der Schaltungen.
Bizacki P., Paliwoda G., Skomorowski M.: Interakcyjny system do projektowania i analizy układów logicznych INFORMATYKA 1992, nr 11, s. 5 Charakterystyka interakcyjnego systemu do projektowania i analizy układów logicznych, przeznaczonego do realizacji na komputerach klasy IBM PC.	**Bizacki P., Paliwoda G., Skomorowski M.: An interactive system for designing and analysis of logical circuits** INFORMATYKA 1992, No. 11, p. 5 Characteristics of an interactive system for designing and analysis of logical circuits, devoted for realization on computers of the IBM PC class.	**Bizacki P., Paliwoda G., Skomorowski M.: Interaktives System für Projektierung und Analyse der logischen Schaltungen** INFORMATYKA 1992, Nr. 11, S. 5 Eine Charakteristik des interaktiven Systems für Projektierung und Analyse von logischen Schaltungen, das für Realisierung auf Computern von IBM PC-Klasse bestimmt ist.
Michalewski E.: DIANA-9 – pakiet wspomaganej komputerowo analizy diagnostycznej i projektowania struktur organizacyjnych INFORMATYKA 1992, nr 11, s. 9 Charakterystyka pakietu DIANA-9 na komputery klasy IBM PC do wspomaganej komputerowo analizy diagnostycznej i projektowania struktur organizacyjnych.	**Michalewski E.: DIANA-9 – a package for commputer assisted diagnostic analysis and designing of organizational structures** INFORMATYKA 1992, No. 11, p. 9 Characteristics of the DIANA-9 package on IBM PC class computers for computer assisted diagnostic analysis and designing of organizational structures.	**Michalewski E.: DIANA-9 – ein Paket für computerunterstützte Diagnostikanalyse und Projektierung von Organisationsstrukturen** INFORMATYKA 1992, Nr. 11, S. 9 Eine Charakteristik des DIANA-9-Pakets auf Computern von IBM PC-Klasse für computerunterstützte Diagnostikanalyse und Projektierung von Organisationsstrukturen.
Szewczyk A.: Diagnostyka systemów informacyjnych (1) INFORMATYKA 1992, nr 11, s. 12 Pierwsza część charakterystyki metod diagnozowania systemów informacyjnych zawierająca propozycję języka opisu takich systemów.	**Szewczyk A.: Information systems diagnostics (1)** INFORMATYKA 1992, No. 11, p. 12 First part of characteristics of information systems diagnosing methods, which includes a proposal of a description language for such systems.	**Szewczyk A.: Diagnostik von Informationssystemen (1)** INFORMATYKA 1992, Nr. 11, S. 12 Erster Teil einer Charakteristik von Methoden des Informationssystemediagnosierens, der ein Vorschlag für Sprache zum Beschreiben eines solchen Systems enthält.
Krajewski R.: Dlaczego NIS? INFORMATYKA 1992, nr 11, s. 19 Omówienie mechanizmu działania oraz sposobów wykorzystania sieciowej usługi informacyjnej systemu NIS.	**Krajewski R.: Why NIS?** INFORMATYKA 1992, No. 11, p. 19 Discussion on operation mechanism and network information service application methods of the NIS system.	**Krajewski R.: Warum NIS?** INFORMATYKA 1992, Nr 11, S. 19 Eine Besprechung des Betriebsmechanismus und Anwendugsmethoden von Netzinformationsdienstleistung des NIS-Systems.
Orylska J.: Dydaktyka informatyki w uczelniach rolniczych INFORMATYKA 1992, nr 11, s. 22 Krytyczne uwagi na temat aktualnego stanu nauczania informatyki w polskich uczelniach rolniczych oraz propozycja zmian.	**Orylska J.: Data processing didactics in agricultural high schools** INFORMATYKA 1992, No. 11, p. 22 Critical remarks on actual situation of data processing teaching in polish agricultural high schools and proposals for changes.	**Orylska J.: Informatikdidaktik in landwirtschaftlichen Hochschulen** INFORMATYKA 1992, Nr. 11, S. 22 Kritische Bemerkungen zum Thema des jetzigen Standes von Informatikunterricht in polnischen landwirtschaftlichen Hochschulen und Vorschläge der Veränderungen.
Winkler T.: Środowisko systemów CAD INFORMATYKA 1992, nr 11, s. 25 Charakterystyka rozwoju środowiska programowego współczesnych systemów CAD, wspomagającego procesy towarzyszące projektowaniu.	**Winkler T.: CAD systems environment** INFORMATYKA 1992, No. 11, p. 25 Characteristics of the development of contemporary CAD systems environment, which assist processes connected with design work.	**Winkler T.: Umwelt von CAD-Systemen** INFORMATYKA 1992, Nr. 11, S. 25 Eine Charakteristik der Softwareumwelt von heutigen CAD-Systemen, die die mit Projektierung verbundene Prozesse unterstützt.

Produkty sieciowe LAN i WAN

Gateway communications, inc.



BEZKONKURENCYJNE PRODUKTY
SIECI KOMPUTEROWYCH NAGRADZANE PRZEZ:

PC MAGAZINE, LAN MAGAZINE, INFO WORLD

OFERUJE AUTORYZOWANY DYSTRYBUTOR:

MIKROB

P.W.P.T. MIKROB SP. Z O.O.

20-346 Lublin, ul. Długa 5

Tel. (0-81) 420-61, faks 415-43, teleks 643776 MIKRO PL

0/15/91

Na życzenie wysyłamy katalog produktów. Atrakcyjny program współpracy dla dealer'ów.

JUNISOFTEX Sp. z o.o.

44-100 GLIWICE ul. Konstytucji 11,
tel.-faks 31-75-10, 31-90-81 do 88 w. 250, 272, 282, teleks 036233

JUNISOFTEX – to firma z tradycjami i najdłuższymi doświadczeniami w eksploatacji wielodostępnych systemów komputerowych Novell w kraju.

JUNISOFTEX – to najstarszy w kraju wykonawca własnych wielodostępnych systemów pracujących w sieci NetWare firmy Novell.

JUNISOFTEX – to autoryzowany reseller amerykańskiej firmy Novell.

JUNISOFTEX – to dostawca zintegrowanych systemów komputerowych działających w kilkudziesięciu firmach obejmujących w każdym przedsiębiorstwie, niezależnie od formy własności i dziedziny gospodarki: finanse, majątek obrotowy, majątek trwały i nietrwały, sprzedaż, techniczne przygotowanie produkcji oraz kadry-płace.

JUNISOFTEX – to dostawca i wykonawca autoryzowanych sieci lokalnych firmy Novell.

JUNISOFTEX – to dostawca sprawdzonego sprzętu komputerowego renomowanych firm ALR, IBM, TEAM, DIGILAB, EPSON, MANNESMANN TALLY.

JUNISOFTEX – to wyłączny i autoryzowany dystrybutor komputerów i terminali firmy DIGILAB na Śląsku.

JUNISOFTEX – to dostawca i wykonawca okablowania lokalnych sieci komputerowych ARCNET, ETHERNET na kablach zwykłych i światłowodowych oraz na dwużyłowych kablach telefonicznych w przypadku stosowania oprogramowania NetWare ACCESS SERVER firmy Novell.

JUNISOFTEX – to nauczyciel, który chętnie podzieli się swoją wiedzą na organizowanych kursach w swojej szkole informatycznej – w tym nauczy Cię: podstaw informatyki, obsługi sprzętu komputerowego, eksploatacji własnych systemów informatycznych, edytorów tekstu: CHiWRITER, WordPerfect, arkuszy kalkulacyjnych QPRO v. 3.0, LOTUS 1-2-3 itp.

JUNISOFTEX – to doradca w zakresie komputerowych metod organizacji i eksploatacji systemów komputerowych.

JUNISOFTEX – to strażnik postępu i nowoczesności w Twojej firmie.

JUNISOFTEX – to gwarancja niezawodności, rzetelności i terminowości.

JUNISOFTEX – to Twój doradca i partner, któremu możesz zaufać, który Cię nigdy nie zawiedzie.

JUNISOFTEX – to partner, który Cię wysłucha i zawsze pomoże podjąć dobrą decyzję.

Jeżeli chcesz

- ☆ zastosować najnowocześniejsze systemy komputerowe oraz lokalne sieci komputerowe,
- ☆ zreorganizować, usprawnić i szybko skomputeryzować swoje przedsiębiorstwo,
- ☆ lepiej wykorzystać już istniejące w Twoim przedsiębiorstwie komputery,
- ☆ wymienić swoje niefortunnie zakupione oprogramowanie,
- ☆ pozbyć się problemów eksploatacyjnych, upadających systemów, nie działających komputerów,
- ☆ przeszkolić załogę swojego przedsiębiorstwa,
- ☆ dobrze zainwestować swoje pieniądze i podnieść rangę swojego przedsiębiorstwa,
- ☆ w przyszłości mieć dostęp do krajowej i światowej sieci komputerowej,

TO ZGŁOŚ SIĘ DO NAS!

0/17/91